

ArrayList

Hemos visto como crear listas de números con una cantidad máxima, lo que llamamos *arrays*, si poníamos, por ejemplo: `int[] nums = new int[10]`; esa lista va a tener como máximo 10 números, sin la posibilidad de añadir un onceavo pase lo que pase, los *ArrayList* son *arrays infinitos*, puedes meterles tantos número como quieras, y el propio programa va añadiéndole memoria a medida que le vamos metiendo cosas.



```
1 import java.util.ArrayList;
2
3 public class Pruebas {
4
5     static ArrayList cosas = new ArrayList();
6
7     public static void main (String[] args) {
8
9         cosas.add(5);
10        cosas.add('c');
11        cosas.add("Hola");
12
13
14        System.out.println(cosas.get(1));
15    }
16
17 }
```

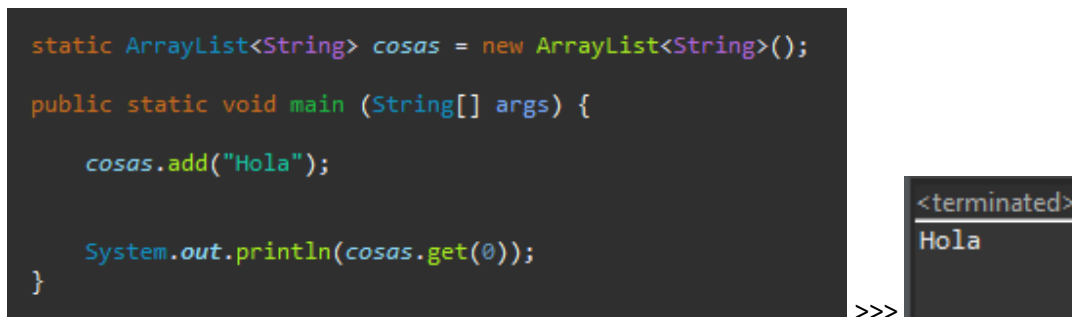
>>>

Problems @ Java

<terminated> Pruebas [

c

Para poder usar un *ArrayList* tenemos que poner **`import java.util.ArrayList;`** antes de declarar la clase, y si nos encontramos en un paquete, siempre después de la línea *package nombreDelPaquete;* Otra diferencia de los *ArrayList* es que estos pueden contener cualquier tipo de variable, como veis le he añadido un *int*, un *char* y un *String* y funciona perfectamente, aun así podemos hacer que los *ArrayList* solo acepten un tipo de dato, pero este tipo ha de ser uno *no primitivo*, es decir, un objeto.



```
static ArrayList<String> cosas = new ArrayList<String>();

public static void main (String[] args) {

    cosas.add("Hola");

    System.out.println(cosas.get(0));
}
```

>>>

<terminated>

Hola

Para ello justo después de *ArrayList*, pondremos: *<tipoDeObjeto>*, os dejo en la siguiente página todos los métodos que tiene *ArrayList*:

Nombre	Dato devuelto	Descripción
add(E e)	boolean	Añade el elemento e al final de la lista.
add(int i, E e)	void	Añade el elemento e en la posición i .
clear()	void	Borra la lista entera.
clone()	Object	Devuelve una copia de la lista.
contains(Object o)	boolean	Indica si el objeto o se encuentra en la lista.
get(int i)	E	Devuelve el elemento en la posición i .
indexOf(Object o)	int	Devuelve el primer índice de o , si no está, devuelve -1.
lastIndexOf(Object o)	int	Devuelve el último índice de o , si no está, devuelve -1.
isEmpty()	boolean	Devuelve si la lista está vacía o no.
remove(int i)	boolean	Elimina el elemento en la posición i .
remove(Object o)	boolean	Elimina el objeto o de la lista.
set(int i, E e)	E	Cambia el elemento de la posición i por e .
size()	int	Devuelve el tamaño de la lista.
toArray()	Object[]	Devuelve la lista convertida en <i>array</i> .

Aclaración: Un elemento es un tipo de dato tanto *primitivo* como un *Objeto*.

Aquellos que ponen que devuelven un *boolean* pero lo único que hacen es realizar acciones como añadir un elemento a la lista o eliminar un elemento de la misma, éstos métodos devuelven *true* si han podido realizar la operación con éxito, y *false* en caso contrario.