ArrayList con ejemplos

Declaración

De forma general un ArrayList en Java se crea de la siguiente forma:

```
ArrayList nombreArray = new ArrayList();
```

Esta instrucción crea el ArrayList nombreArray vacío.

Un arrayList declarado así puede contener objetos de cualquier tipo.

Una alternativa a esta declaración es indicar el tipo de objetos que contiene. En este caso, el array solo podrá contener objetos de ese tipo:

```
ArrayList<tipo> nombreArray = new ArrayList<tipo>();
```

Declaración

```
ArrayList<tipo> nombreArray = new ArrayList<tipo>();
```

tipo debe ser una clase. Indica el tipo de objetos que contendrá el array. No se pueden usar tipos primitivos. Para un tipo primitivo se debe utilizar su clase envolvente.

Por ejemplo:

```
ArrayList<Integer> numeros = new ArrayList<Integer>();
```

Crea el array *numeros* de enteros.

Ejemplo: Añadimos 10 Elementos en el ArrayList

```
for (int i=1; i<=10; i++){

    nombreArrayList.add("Elemento "+i); }
```

Algunos MÉTODOS de ARRAYLIST

MÉTODO	DESCRIPCIÓN
size()	Devuelve el número de elementos (int)
add(X)	Añade el objeto X al final. Devuelve true.
add(posición, X)	Inserta el objeto X en la posición indicada.
get(posicion)	Devuelve el elemento que está en la posición indicada.
remove(posicion)	Elimina el elemento que se encuentra en la posición indicada. Devuelve el elemento eliminado.
remove(X)	Elimina la primera ocurrencia del objeto X. Devuelve true si el elemento está en la lista.
clear()	Elimina todos los elementos.
set(posición, X)	Sustituye el elemento que se encuentra en la posición indicada por el objeto X. Devuelve el elemento sustituido.
contains(X)	Comprueba si la colección contiene al objeto X. Devuelve true o false.
indexOf(X)	Devuelve la posición del objeto X. Si no existe devuelve -1

RECORRER UN ARRAYLIST

De la manera clásica con un bucle for y el método get(i):

```
for(int i = 0;i<array.size();i++){
            System.out.println(array.get(i));
}
```

Con un bucle foreach:

```
for(Object o: nombreArray){
            System.out.println(o);
}
```

RECORRER UN ARRAYLIST

Utilizando un **objeto Iterator**.

La ventaja de utilizar un Iterador es que no necesitamos indicar el tipo de objetos que contiene el array.
Iterator tiene como métodos:

- **hasNext**: devuelve true si hay más elementos en el array.
- **next**: devuelve el siguiente objeto contenido en el array.

Ejemplo:

```
ArrayList<Integer> numeros = new ArrayList<Integer>();
…..
//se insertan elementos
.....
Iterator it = numeros.iterator(); //se crea el iterador it para el array numeros
while(it.hasNext())                    //mientras queden elementos
        System.out.println(it.next());  //se obtienen y se muestran
```

La API Collections Genérica