Iteradores

La interfaz Iterator<E>

Representa un iterador sobre una colección. Permiten recorrer esa colección elemento a elemento. Además permiten eliminar de la colección el útltimo elemento que nos ha devuelto el iterador.

Un objeto de una clase que implemente la interfaz Iterator, es un iterador (no pueden instanciarse objetos Iterator directamente, ya que es una interfaz).

Para generarlos, recurriremos al método iterator() que nos proporcionan las clases que implementan la interfaz iterable (entre otras todas las listas y conjuntos)

Métodos definidos en Iterator

boolean hasNext(): Devuelve true si hay más elementos disponibles en el iterador y false si no hay más.

<E> next(): Devuelve el siguiente objeto disponible en el iterador (el objeto será de tipo E, recordad los genéricos para no tener que estar casteando continuamente de Object).

void remove(): Elimina el último objeto que ha devuelto el iterador DE LA COLECCIÓN SUBYACENTE, es decir la elimina del origen de datos.

for each utiliza un Iterator

Cuando utilizamos el método for each, en realidad se está construyendo un iterador por el que se va recorriendo la colección correspondiente.

Podremos utilizar un for each o un iterador indistintamente SALVO que queramos tener la posibilidad de eliminar elementos de la colección. El único que nos permite esto es Iterator con su método remove()