Recorrer colecciones

DigitalHouse >
Coding School

Certified Tech Developer
The Ultimate Degree

Índice

1. for / while
2. Iterator
3. for each

DigitalHouse> Coding School

“

En esta presentación veremos tres maneras de recorrer una colección.

”

1 | for / while

DigitalHouse> Coding School

for / while

Una manera de recorrer una colección es a través de un ciclo for:


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

size

Será igual a 5. Ya que tenemos 5 nombres almacenados en la colección.

get

Con get iremos obteniendo el valor de cada una de las posiciones.

for / while

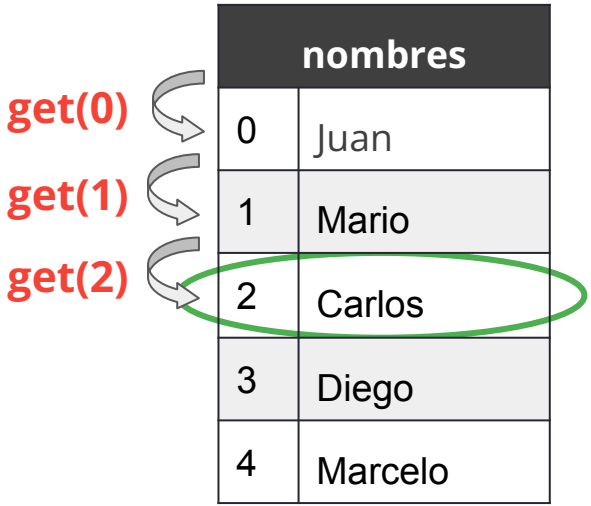
Otra manera muy similar es hacerlo con un ciclo while. Es muy útil cuando necesitamos cortar el ciclo antes de recorrer todos sus elementos.

	nombres
0	Juan
1	Mario
2	Carlos
3	Diego
4	Marcelo

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

for / while

En el siguiente ejemplo, necesitamos encontrar a “Carlos”, con lo cual, una vez hallado, podemos salir del bucle para no seguir recorriendo innecesariamente.




```
int i = 0;
boolean encontrado = false;
while( i < nombres.size() && !encontrado)
{
   if(nombres.get(i) == "Carlos")
       encontrado = true;
   System.out.println(nombres.get(i));
   i++;
}
```

for / while

Para poder recorrer una colección con un ciclo for o while, podemos observar que necesitamos de los métodos **size()** y **get()**

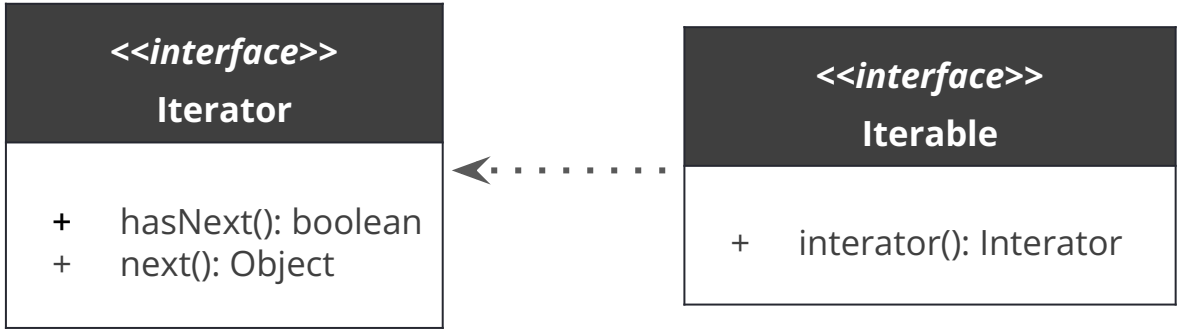
No todas las colecciones poseen estos métodos con lo cual **no podremos utilizar estas opciones en algunas colecciones**.

Solo podemos utilizar estas opciones de recorrido con las **List**, es decir, con **ArrayList** y **LinkedList**.

2 | Iterator

Interator

En Java las colecciones implementan la interface Iterable, lo que obliga a implementar el método **iterator()**.



Iterator() devolverá un objeto del tipo Iterator, donde mediante los métodos **hasNext()** y **next()** podremos recorrer la colección.

Iterator

Utilizar el método iterator() es otra manera de recorrer las colecciones y podremos hacerlo en todos los tipos de colecciones.

next()

nos devuelve el próximo elemento.

```
Iterator iterator = nombres.iterator();
while(iterator.hasNext()){
   System.out.println(iterator.next());
}
```

hasNext()

nos indica si hay o no hay todavía elementos en la colección.

3 | for each

for each

Muchos lenguajes poseen una manera simple y elegante de recorrer una colección a través de los **ciclos for each**. Desde la versión 1.5 de Java se incluyó esta sencilla forma de recorrer las colecciones.

nombres	
0	Juan
1	Mario
2	Carlos
3	Diego
4	Marcelo

Por cada **objeto** en la colección **nombres** traerlo y colocarlo en el objeto **nombre.**

```
for(Object nombre: nombres){
   System.out.println(nombre);
}
```


DigitalHouse>
Coding School