

Percorrendo Coleções

DigitalHouse>



**Certified Tech
Developer**

The Ultimate Degree

Índice

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

1 | For / while

for

Uma das maneiras de acessar uma coleção é através de um ciclo **for**:

nomes	
get(0)	0 Juan
get(1)	1 Mario
get(2)	2 Carlos
get(3)	3 Diego
get(4)	4 Marcelo

.size()

Será igual a 5. Já que temos 5 nomes armazenados na coleção.

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

.get()

Com get obteremos o valor de cada uma das posições.

while

Outra maneira, é através do ciclo **while**. Essa forma é muito útil quando precisamos parar o ciclo antes de acessar todos os elementos.

	nomes	
get(0)	0	Juan
get(1)	1	Mario
get(2)	2	Carlos
get(3)	3	Diego
get(4)	4	Marcelo

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

for / while

No exemplo a seguir, precisamos encontrar "Carlos", com o qual, uma vez encontrado, podemos sair do loop para não continuar percorrendo desnecessariamente.

	nomes	
get(0)	0	Juan
get(1)	1	Mario
get(2)	2	Carlos
	3	Diego
	4	Marcelo

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

for / while

Para poder percorrer uma coleção com um ciclo for ou while, podemos observar que necessitamos dos métodos **size()** e **get()**.



Nem todas as coleções têm esses métodos, portanto, não poderemos usar essas opções em algumas coleções.

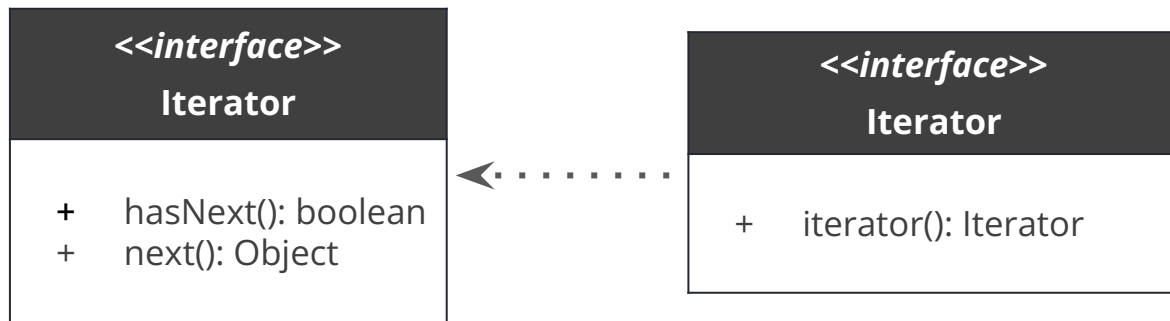


Podemos utilizar esses métodos apenas com **List**, ou seja, com **ArrayList** e **LinkedList**.

2 | Iterator

Iterator

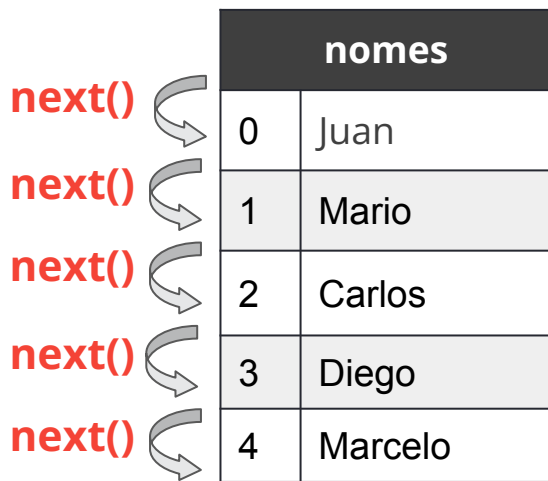
Em java, as coleções implementam a interface Iterator, que obriga a implementar o método **iterator()**.



iterator() devolverá um objeto do tipo **Iterator**, onde mediante os métodos **hasNext()** e **next()** podemos acessar a coleção.

Iterator

Utilizar o método `iterator()` é uma outra maneira de termos acesso às coleções e podemos usá-lo em todos os tipos de coleção.



nomes	
0	Juan
1	Mario
2	Carlos
3	Diego
4	Marcelo

next()

nos devolve o próximo elemento

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

hasNext()

indica se há ou não elementos na coleção.

3 | **for each**


for each

Muitas linguagens têm uma maneira simples e elegante de percorrer uma coleção para cada loop. Desde a versão **1.5 do Java**, esta maneira simples de navegar pelas coleções foi incluída.



nomes	
0	Juan
1	Mario
2	Carlos
3	Diego
4	Marcelo

Para cada objeto na coleção de nomes, **traga-o** e **coloque-o** no nome do objeto.



```
for(Object nome: nomes){  
    System.out.println(nome);  
}
```

DigitalHouse>
Coding School