Introducing the forEachRemaining() in Iterator.

This lesson explains the forEachRemaining() method, which was introduced in the iterator class in Java 8.

We'll cover the following

`forEachRemaining()` in `Iterator`

`Iterator` is an interface available in the `Collections` framework in `java.util` package. It is used to iterate a collection of objects. This interface has four methods, as shown in the below image. Before, Java 8 the `forEachRemaining()` method did not exist.

Method Summary

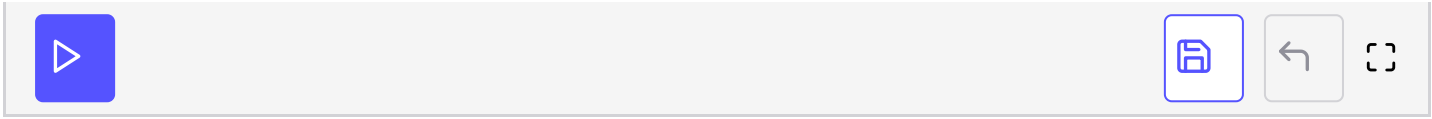
All Methods | Instance Methods | Abstract Methods | Default Methods

Modifier and Type	Method and Description
default void	forEachRemaining(Consumer<? super E> action) Performs the given action for each remaining element until all elements have been processed or the action throws an exception.
boolean	hasNext() Returns true if the iteration has more elements.
E	next() Returns the next element in the iteration.
default void	remove() Removes from the underlying collection the last element returned by this iterator (optional operation).

Below is a simple program to iterate a list using iterator before Java 8.

```
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;

public class IteratorDemo {

    public static void main(String args[]) {

        List<String> fruits = new ArrayList<>();
        fruits.add("Apple");
        fruits.add("Banana");
        fruits.add("Grapes");
        fruits.add("Orange");

        Iterator<String> iterator = fruits.iterator();
```

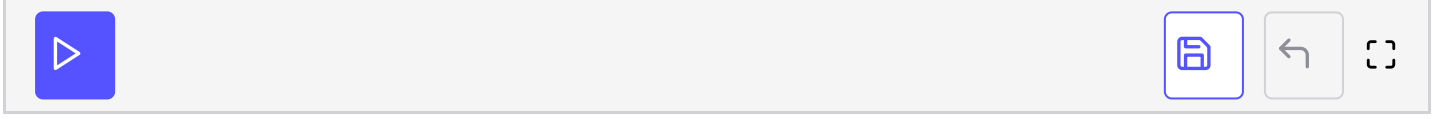
```
        while (iterator.hasNext()) {
            System.out.println(iterator.next());
        }

    }
}
```



As you can see in the above example requires a while loop in order to iterate through the input list via an `Iterator`. To avoid this, the `forEachRemaining()` method was introduced in Java 8. This method takes in a `Consumer` instance as a parameter.

As you have seen in the `Consumer` interface lesson that it taken in a parameter and does not return anything. This is what we require for our iterator. Below is the same example shown above, but, this time, we are using the `forEachRemaining()` method.

```
import java.util.ArrayList;
import java.util.Iterator;
import java.util.List;

public class IteratorDemo {

    public static void main(String args[]) {

        List<String> fruits = new ArrayList<>();
        fruits.add("Apple");
        fruits.add("Banana");
        fruits.add("Grapes");
        fruits.add("Orange");

        Iterator<String> iterator = fruits.iterator();

        iterator.forEachRemaining((fruit) -> System.out.println(fruit));
    }
}
```



Therefore, the main purpose of introducing the `forEachRemaining()` method was to make the iteration code more concise and readable.

In the next lesson, we will discuss improvements in `Map` API.