java.util

Interface Iterator<E>

Type Parameters:

E - the type of elements returned by this iterator

All Known Subinterfaces:

ListIterator<E>, XMLEventReader

All Known Implementing Classes:

BeanContextSupport.BCSIterator, EventReaderDelegate, Scanner

```
public interface Iterator<E>
```

An iterator over a collection. `Iterator` takes the place of `Enumeration` in the Java Collections Framework. Iterators differ from enumerations in two ways:

- Iterators allow the caller to remove elements from the underlying collection during the iteration with well-defined semantics.
- Method names have been improved.

This interface is a member of the Java Collections Framework.

Since:

1.2

See Also:

`Collection`, `ListIterator`, `Iterable`

Method Summary

Methods

Modifier and Type	Method and Description
`boolean`	**`hasNext`**`()` Returns `true` if the iteration has more elements.
`E`	**`next`**`()` Returns the next element in the iteration.
`void`	**`remove`**`()` Removes from the underlying collection the last element returned by this iterator (optional operation).

Method Detail

hasNext

```
boolean hasNext()
```

Returns `true` if the iteration has more elements. (In other words, returns `true` if `next()` would return an element rather than throwing an exception.)

Returns:

`true` if the iteration has more elements

next

`E next()`

Returns the next element in the iteration.

Returns:

the next element in the iteration

Throws:

`NoSuchElementException` - if the iteration has no more elements

remove

`void remove()`

Removes from the underlying collection the last element returned by this iterator (optional operation). This method can be called only once per call to `next()`. The behavior of an iterator is unspecified if the underlying collection is modified while the iteration is in progress in any way other than by calling this method.

Throws:

`UnsupportedOperationException` - if the `remove` operation is not supported by this iterator

`IllegalStateException` - if the `next` method has not yet been called, or the `remove` method has already been called after the last call to the `next` method