Class CircularList.CircularListIterator<E>

```
java.lang.Object
  CircularList.CircularListIterator<E>
```

All Implemented Interfaces: java.util.Iterator<E>, java.util.ListIterator<E> **Enclosing class:**CircularList<E>

```
public class CircularList.CircularListIterator<E>extends
java.lang.Objectimplements java.util.ListIterator<E>
```

This class provides an iterator for this CircularList.

Author: Bryan Davis

Field Summary	
private int	**nextIndex** The index of nextNode
private CircularNode<E>	**nextNode** The next node that this iterator points to
private CircularNode<E>	**previousNode** The last node returned by next() or previous()

Constructor Summary

CircularList.CircularListIterator(int index) Construct a CircularListIterator whose nextNode points at the node with the given index

Method Summary	
void	**add**(E e) This method is not supported and must throw an UnsupportedOperationException if called.
boolean	**hasNext**() Return true since circular linked lists nodes always have a next node.
boolean	**hasPrevious**() Return true since circular linked lists nodes always have a previous node.
E	**next**() Returns the next element in the list.
int	**nextIndex**() Returns the index of the next node
E	**previous**() Returns the previous element in the list.
int	**previousIndex**() Returns the index of the previous node
void	**remove**() This method is not supported and must throw an UnsupportedOperationException if called.
void	**set**(E e) This method is not supported and must throw an UnsupportedOperationException if called.

Methods inherited from class java.lang.Object
clone, equals, finalize, getClass, hashCode, notify, notifyAll, toString, wait, wait, wait

Field Detail

nextNode

```
private CircularNode<E> nextNode
```

The next node that this iterator points to

---

nextIndex

`private int` **`nextIndex`**

The index of nextNode

---

previousNode

`private CircularNode<E>` **`previousNode`**

The last node returned by next() or previous()

Constructor Detail

CircularList.CircularListIterator

`public` **`CircularList.CircularListIterator`**`(int index)`

Construct a CircularListIterator whose nextNode points at the node with the given index

Parameters:index - the index of the node that this iterator's nextNode will point to

Method Detail

add

`public void` **`add`**`(E e)`

This method is not supported and must throw an UnsupportedOperationException if called.

Specified by:add in interface java.util.ListIterator<E> **Throws:**
java.lang.UnsupportedOperationException - Throw this error whenever called.

---

hasNext

`public boolean` **`hasNext`**`()`

Return true since circular linked lists nodes always have a next node. WARNING! This function should not be used as a condition to iterate a list since it may result in an infinite loop.

Specified by:hasNext in interface java.util.Iterator<E>**Specified by:**hasNext in interface

java.util.ListIterator<E>

---

hasPrevious

`public boolean hasPrevious()`

Return true since circular linked lists nodes always have a previous node. WARNING! This function should not be used as a condition to iterate a list since it may result in an infinite loop.

Specified by:hasPrevious in interface java.util.ListIterator<E>

---

next

`public E next()`

Returns the next element in the list. This method may be called repeatedly to iterate through the list, or intermixed with calls to previous to go back and forth. (Note that alternating calls to next and previous will return the same element repeatedly.) The next element of the last node of the list is the first node's element.

Specified by:next in interface java.util.Iterator<E>**Specified by:**next in interface java.util.ListIterator<E>

---

nextIndex

`public int nextIndex()`

Returns the index of the next node

Specified by:nextIndex in interface java.util.ListIterator<E>

---

previous

`public E previous()`

Returns the previous element in the list. This method may be called repeatedly to iterate through the list, or intermixed with calls to next to go back and forth. (Note that alternating calls to next and previous will return the same element repeatedly.) The previous element of the first node of the list is the last node's element.

Specified by:previous in interface java.util.ListIterator<E>

---

previousIndex

`public int previousIndex()`

Returns the index of the previous node

Specified by:previousIndex in interface java.util.ListIterator<E>

---

remove

```
public void remove()
```

This method is not supported and must throw an UnsupportedOperationException if called.

Specified by:remove in interface java.util.Iterator<E>**Specified by:**remove in interface java.util.ListIterator<E> **Throws:** java.lang.UnsupportedOperationException - Throw this error whenever called.

---

set

```
public void set(E e)
```

This method is not supported and must throw an UnsupportedOperationException if called.

Specified by:set in interface java.util.ListIterator<E> **Throws:** java.lang.UnsupportedOperationException - Throw this error whenever called.
