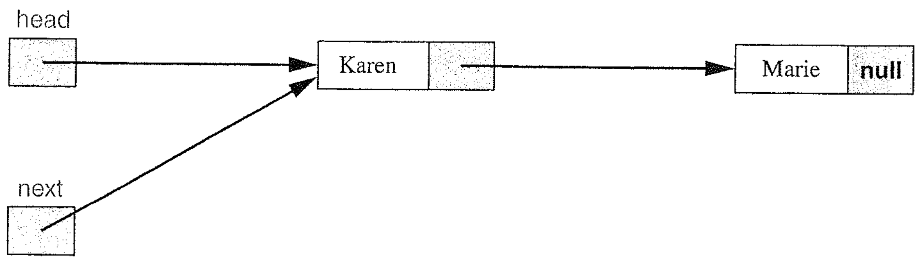
Iterator Constructor/ next

Figure 7.7 | The contents of the **next** field in the SinglyLinkedListIterator class just after the SinglyLinkedListIterator's constructor is called.



do anything after a return, so we save next.element before advancing next, and then we return (a reference to) the saved element. Here is the definition:

```
public E next( )
{
        E theElement = next.element;
        next = next.next;   // rightmost next is field in Entry class
        return theElement;
} // method next
```

Now that we have a SinglyLinkedListIterator class, we can work on the problem of iterating through a SinglyLinkedList object. First, we have to associate a SinglyLinkedListIterator object with a SinglyLinkedList object. The iterator( ) method in the SinglyLinkedList class creates the necessary connection:

```
/**
 * Returns a SinglyLinkedListIterator object to iterate over this
 * SinglyLinkedList object.
 *
 */
public Iterator<E> iterator( )
{
        return new SinglyLinkedListIterator( );
} // method iterator
```

The value returned is a (reference to a) SinglyLinkedListIterator. The specified return type has to be Iterator<E> because that is what the iterator( ) method in the Iterator interface calls for. Any class that implements the Iterator interface—such as SinglyLinkedListIterator—can be the actual return type.

With the help of this method, a user can create the appropriate iterator. For example, if myLinked is a SinglyLinkedList object of Boolean elements, we can do the following:

```
Iterator<Boolean> itr = myLinked.iterator( );
```