

## Re: CurrentElement->next = CurrentElement->next->next (UNDEFINED?)

*Source:* [http://coding.derkeiler.com/Archive/C\\_CPP/comp.lang.c/2005-03/0972.html](http://coding.derkeiler.com/Archive/C_CPP/comp.lang.c/2005-03/0972.html)

---

*From:* Andrey Tarasevich ([andreytarasevich\\_at\\_hotmail.com](mailto:andreytarasevich_at_hotmail.com))

*Date:* 03/09/05

Date: Tue, 08 Mar 2005 16:45:07 -0800

Deniz Bahar wrote:

```
> ...
> I'm working with a single linked list and want to delete elements by
> searching through the list (starting form the HEAD) then finding the
> element, then doing the following:
>
> NewElement = CurrentElement->next;
> CurrentElement->next = NewElement->next->next;
> free(NewElement);
>
> Does the double ->next invoke undefined behaviour (sequence points
> etc)?
> ...
```

No. Firstly, you are not accessing the same object twice. 'CurrentElement->next' as an lvalue is neither 'NewElement' nor 'NewElement->next' nor 'NewElement->next->next'. There's no problem with sequence points here.

However, the code itself doesn't do what it is supposed to do. Doing

```
NewElement = CurrentElement->next;
CurrentElement->next = NewElement->next->next;
```

will exclude `_two_` consecutive elements from the list (both 'NewElement' and 'NewElement->next'), not one. Apparently, this is not what you wanted to do. You probably intended to do either

```
NewElement = CurrentElement->next;
CurrentElement->next = NewElement->next;
```

or

```
NewElement = CurrentElement->next;
CurrentElement->next = CurrentElement->next->next;
```

comp.lang.c: Re: CurrentElement->next = CurrentElement->next->next (UNDEFINED?)

That would exclude only one element from the list.

Now in that last version the value of 'CurrentElement->next' is accessed twice (once to read it and once to modify it), which might rise the question about sequence points etc. However, in this case everything is fine too. The old value of 'CurrentElement->next' is read for the sole purpose of determining its new value.

--

Best regards,  
Andrey Tarasevich