

# Re: pointer to void property

---

*Source:* [http://coding.derkeiler.com/Archive/C\\_CPP/comp.lang.c/2005-09/msg01085.html](http://coding.derkeiler.com/Archive/C_CPP/comp.lang.c/2005-09/msg01085.html)

---

- *From:* "Mabden" <[mabden@xxxxxxxxxxxxxxxx](mailto:mabden@xxxxxxxxxxxxxxxx)>
  - *Date:* Mon, 12 Sep 2005 23:48:43 GMT
- 

"Old Wolf" <[oldwolf@xxxxxxxxxxxxxxxx](mailto:oldwolf@xxxxxxxxxxxxxxxx)> wrote in message  
[news:1126566243.859723.271250@xx](mailto:news:1126566243.859723.271250@xx)

> Irrwahn Grausewitz wrote:

>>

>> If it's safe to convert a p-to-sometype to p-to-void, AND it's safe

>> to convert a p-to-void to p-to-char, THEN it's safe to convert a

>> p-to-int to p-to-void to p-to-char.

>>

>> What a typing mess...

>

> The OP's original question was (rephrased):

>

> Can this p-to-char be used to access the representation of the

> "sometype" that p-to-sometype was pointing to?

>

> which doesn't seem to have been answered yet on this thread

> (everyone was distracted by the casting issue). I'm not sure

> of the answer personally.

>

Well, even if you do so, you would have to account for (ie: guess) byte alignments that might change between compiler versions, or CPU changes (32bit vs 64bit currently), or OS changes (versions), or OS changes (endian issues), or OS changes (devices like PDAs), or OS changes (you get the idea).

—  
Mabden

- 
- *Follow-Ups:*
    - ◆ ***Re: pointer to void property***
      - ◇ *From:* Irrwahn Grausewitz

- *References:*

Re: pointer to void property

- ◆ *pointer to void property*
  - ◇ *From: aegis*
- ◆ *Re: pointer to void property*
  - ◇ *From: Irrwahn Grausewitz*
- ◆ *Re: pointer to void property*
  - ◇ *From: aegis*
- ◆ *Re: pointer to void property*
  - ◇ *From: Irrwahn Grausewitz*
- ◆ *Re: pointer to void property*
  - ◇ *From: Old Wolf*

- Prev by Date: *Re: 0 vs '\0'*
- Next by Date: *Re: Difference of 'also'*
- Previous by thread: *Re: pointer to void property*
- Next by thread: *Re: pointer to void property*
- Index(es):
  - ◆ *Date*
  - ◆ *Thread*