

# Re: strange grammar about volatile and operator overload

---

*Source:* <http://www.tech-archive.net/Archive/VC/microsoft.public.vc.language/2008-01/msg00846.html>

---

- *From:* George <[George@xxxxxxxxxxxxxxxxxxxxxxxxxxxxx](mailto:George@xxxxxxxxxxxxxxxxxxxxxxxxxxxxx)>
  - *Date:* Tue, 29 Jan 2008 19:00:01 -0800
- 

Thanks Igor,

My question is answered.

regards,  
George

"Igor Tandetnik" wrote:

"George" <[George@xxxxxxxxxxxxxxxxxxxxxxxxxxxxx](mailto:George@xxxxxxxxxxxxxxxxxxxxxxxxxxxxx)> wrote in message  
[news:BB2B4E82-3B35-482F-A41B-5E7C60DBDC00@xxxxxxxxxxxxx](mailto:news:BB2B4E82-3B35-482F-A41B-5E7C60DBDC00@xxxxxxxxxxxxx)

But I am not sure whether volatile is consistent to the reference  
binded object. For example, the operator conversion function returns  
reference to some global object and it is marked as volatile now, but  
actually in the definition/declaration of the global object, it is  
not marked as volatile.

That's perfectly OK. Just as you can have a reference-to-const bound to  
an object that wasn't declared const:

```
int x;  
const int& cr = x; // OK  
volatile int& vr = x; // also OK
```

When x is accessed directly, it's not treated as volatile or const. When  
x is accessed via vr reference, it's treated as volatile (meaning  
certain compiler optimizations are suppressed). When x is accessed via  
cr reference, it's treated as const (meaning you can read the value but  
can't change it).

I am not sure about whether I have made  
myself understood about the volatile consistent issue.

Re: strange grammar about volatile and operator overload

No, not really.

Any comments? Is it an issue

Is what an issue?

or just make the global variable in my  
sample more restrictive (volatile is more restrictive than  
non-volatile)?

Yes, volatile is more restrictive.

--

With best wishes,  
Igor Tandetnik

With sufficient thrust, pigs fly just fine. However, this is not  
necessarily a good idea. It is hard to be sure where they are going to  
land, and it could be dangerous sitting under them as they fl