Subject: overloading operators rules
Posted by amando1957 on Wed, 03 Dec 2008 12:31:11 GMT
View Forum Message <> Reply to Message

Hi to all.

There are rules for overloading operators in C++ I'm not sure i got them all. I do not ask for a syntax tutor, its meant so "what makes sense".

- keep track of semantics, so that the operator performs it is assumed to do: the op+ should add anything.
- avoid a given overload provided by the lang: "int % int" is here already.

Anyone can tell me some further rules recommended?
Thank you for any reply.

Martin

Subject: Re: overloading operators rules
Posted by mirek on Wed, 03 Dec 2008 13:45:22 GMT
View Forum Message <> Reply to Message

amando1957 wrote on Wed, 03 December 2008 07:31Hi to all.

There are rules for overloading operators in C++ I'm not sure i got them all. I do not ask for a syntax tutor, its meant so "what makes sense".

- keep track of semantics, so that the operator performs it is assumed to do: the op+ should add anything.

Or, at least, keep the new semantics consistent

Mirek

Subject: Re: overloading operators rules
Posted by captainc on Wed, 03 Dec 2008 22:05:14 GMT
View Forum Message <> Reply to Message

This is more of an 'agreement' that is kept for a project team. I think we should make a Topic++ doc page about it so that we keep all U++ code consistent. Some sort of developer guidelines for contributing would be good.

Subject: Re: overloading operators rules
Posted by piotr5 on Mon, 11 May 2009 14:47:34 GMT
View Forum Message <> Reply to Message

I think a third rule would be to watch out for operator-priority-list. it wouldn't make much sense to define an operator which in combination with other operators needs to be put into brackets all the time. for example in boolean logic "*" is used for and, "+" is used for or, even though the two could be used the other way around, simply because it is more convenient to have a tighter binding between 2 things which live in an and-condition. (for example (a&b)|(c&d) is more natural for us to understand than (a|b)&(c|d)...)

another interesting rule which stl seems to obey is that some operators must have symmetric parameters-types. some algoritms of stl are only defined when both parameters of the operator are of the same type. I don't remember where I had this problem though, but I think it had something to do with the "<" operator...