Subject: <<= and <<
Posted by Balage on Sun, 17 Dec 2006 22:12:50 GMT
View Forum Message <> Reply to Message

What's the difference between:

somebutton << THISBACK(Press);

and

somebutton <<= THISBACK(Press);

?

Subject: Re: <<= and <<
Posted by mirek on Sun, 17 Dec 2006 22:48:04 GMT
View Forum Message <> Reply to Message

<<= sets the WhenAction callback

<< adds a new callback to existing one; if there were any callbacks before assigned, they will be called as well.

In fact, "<<" is sort of experiment and is very rarely used.

Note: You cannot use somebutton << THISBAC... because "<<" is method of Callback, not Ctrl. You would have to write

somebutton.WhenAction << THISBACK....

Mirek

Subject: Re: <<= and <<
Posted by Balage on Sun, 17 Dec 2006 22:57:51 GMT
View Forum Message <> Reply to Message

I was just curious, as both worked fine, and produced the same result, I just didn't know what was the diff.

Actually, somebutton << THISBACK(Press) does work!

class Ctrl has these operators:

```
 Callback     operator<<=(Callback action)  { WhenAction = action; return action; }
 Callback&    operator<<(Callback action)   { return WhenAction << action; }
```

I tried this:

somebutton << THISBACK(Press) << THISBACK(Press2) << THISBACK(Press3) << THISBACK(Press);

The handlers were called in this order:
Press, Press2, Press3, Press

So with <<, I can add multiple callbacks. That's nice.

Subject: Re: <<= and <<
Posted by mirek on Sun, 17 Dec 2006 23:16:05 GMT
View Forum Message <> Reply to Message

Yes, you are right.

Mirek

Subject: Re: <<= and <<
Posted by waxblood on Fri, 30 Nov 2007 12:13:24 GMT
View Forum Message <> Reply to Message

Quote:
<<= sets the WhenAction callback

<< adds a new callback to existing one; if there were any callbacks before assigned, they will be called as well.

In fact, "<<" is sort of experiment and is very rarely used

Reading code snippets in documentation '<<=' is always used instead of <<, but isn't the '<<' behaviour much more natural than the first one? Writing code with '<<' allows the programmer to modify callbacks stack of ancestor classes freely being at least a little more confident that those callbacks will always be executed before any one other callbacks in derived classes.
Using always '<<=' precludes this possibility.

I think it would be useful to develop plugin-oriented programs, too.

I'm wondering, given the fact quite nobody uses the '<<' form, would it be possible to convert the

'<<=' behaviour to '<<'?

David

Subject: Re: <<= and <<
Posted by mirek on Fri, 30 Nov 2007 12:52:56 GMT
View Forum Message <> Reply to Message

waxblood wrote on Fri, 30 November 2007 07:13Quote:
<<= sets the WhenAction callback

<< adds a new callback to existing one; if there were any callbacks before assigned, they will be called as well.

In fact, "<<" is sort of experiment and is very rarely used

Reading code snippets in documentation '<<=' is always used instead of <<, but isn't the '<<' behaviour much more natural than the first one? Writing code with '<<' allows the programmer to modify callbacks stack of ancestor classes freely being at least a little more confident that those callbacks will always be executed before any one other callbacks in derived classes.
Using always '<<=' precludes this possibility.

I think it would be useful to develop plugin-oriented programs, too.

Interesting thoughts, yes, something like this is definitely possible path.

I guess using "<<=" is a result of two issues:

1. "<<=" existed long before "<<"
2. In real world applications, single action per callback is simply 99% of cases.

Quote:
I'm wondering, given the fact quite nobody uses the '<<' form, would it be possible to convert the '<<=' behaviour to '<<'?

Well, once again interesting idea... But here I am a bit afraid about changed semantics.

Mirek

Subject: Re: <<= and <<
Posted by waxblood on Thu, 06 Dec 2007 22:44:16 GMT
View Forum Message <> Reply to Message

Quote:Quote:I'm wondering, given the fact quite nobody uses the '<<' form, would it be possible to convert the '<<=' behaviour to '<<'?

Well, once again interesting idea... But here I am a bit afraid about changed semantics.

Mirek

Well, in fact that wasn't a great idea. Here's some better thought:

Quote:2. In real world applications, single action per callback is simply 99% of cases.

I had the suspect of a similar percentage, but now that it is confirmed I think it's time to plan a switch to the '<<'   form. I think these should be the steps to follow:

a) fix the following problem:
writing
somebutton << THISBACK(Press) <<= THISBACK(Press2);
results in a compiling problem, which is right since adding callback Press to just replace it with Press2 with the <<= doesn't make sense. But, if you write
somebutton <<= THISBACK(Press) << THISBACK(Press2);
IMO the compiler shoudn't complain, since replacing callbacks stack with Press and adding then Press2 seems a logic step. In fact the compiler doesn't complain, but at runtime you get no results when pressing somebutton. If this is not a bug, is a bad incongruence at least.

b) start to write new Ultimate++ code with '<<' form.

c) change examples,reference and tutorials (I think this step should be possibly made in one single pass, to avoid generating confusion among users should be just a matter of find & replace (probably replace _all_)

d) place a warning or a suggestion when encountering '<<=', saying in 99% '<<' should be preferred (maybe pointing to an html page explaining way). This should be only a transitional message. Would it be possible to easily turn off the message for upp source?

e) Replace '<<=' in existing code with '<<' (more time consuming).

David

Subject: Re: <<= and <<
Posted by mirek on Fri, 07 Dec 2007 08:25:19 GMT
View Forum Message <> Reply to Message

waxblood wrote on Thu, 06 December 2007 17:44
Quote:2. In real world applications, single action per callback is simply 99% of cases.

I had the suspect of a similar percentage, but now that it is confirmed I think it's time to plan a switch to the '<<'   form.

IMO, you misunderstood me. In 99% cases <<= is OK. Admitedly, in 99% cases of these cases, "<<" is OK as well

Quote:
d) place a warning or a suggestion when encountering '<<=', saying in 99% '<<' should be preferred (maybe pointing to an html page explaining way). This should be only a transitional message. Would it be possible to easily turn off the message for upp source?

IMO, there are about 0.01% of important cases, where "<<=" over "<<" semantics is required...

Mirek

Subject: Re: <<= and <<
Posted by waxblood on Fri, 07 Dec 2007 09:12:53 GMT
View Forum Message <> Reply to Message

luzr wrote on Fri, 07 December 2007 09:25

IMO, you misunderstood me. In 99% cases <<= is OK. Admitedly, in 99% cases of these cases, "<<" is OK as well

For standard apps, '<<=' I'm positive works fine, I just think some problem may arise with general purpose widgets, or 'components'. These should be extensible by nature, and '<<=' doesn't favour extensibility. Having a dynamical feature like callbacks mechanism and preventing to fully exploit it seems like a waste, especially talking about open source software, where you primary rely on re-using classes written by others. If I'd want to write plugin-expandable software (which is quite diffuse nowadays), I think I'd like more '<<' around...

David

Subject: Re: <<= and <<
Posted by mirek on Sat, 08 Dec 2007 06:17:04 GMT
View Forum Message <> Reply to Message

waxblood wrote on Fri, 07 December 2007 04:12luzr wrote on Fri, 07 December 2007 09:25

IMO, you misunderstood me. In 99% cases <<= is OK. Admitedly, in 99% cases of these cases, "<<" is OK as well

For standard apps, '<<=' I'm positive works fine, I just think some problem may arise with general purpose widgets, or 'components'. These should be extensible by nature, and '<<=' doesn't favour extensibility. Having a dynamical feature like callbacks mechanism and preventing to fully exploit it seems like a waste, especially talking about open source software, where you primary rely on re-using classes written by others. If I'd want to write plugin-expandable software (which is quite diffuse nowadays), I think I'd like more '<<' around...

David

OK, so IMO the real result is: Search and replace <<= with << in CtrlCore and CtrlLib, right?

Mirek