Subject: Gcc compile option proposal
Posted by Didier on Thu, 14 Feb 2013 18:33:05 GMT
View Forum Message <> Reply to Message

Hi all,

While doing some corrections for MSC on code developped using GCC I noticed that MSC gives an error when we forget to return a value in a function that requires one.

```
int  myFunction()
{
printf("Hello World !!");
// no return value
}
```
This code does not compile in MSC while it compile perfectly in GCC when using Theide.

I think we should add the following option to GCC config:
-Werror=return-type
With this GCC and MSC have the same compilation behaviour.

Subject: Re: Gcc compile option proposal
Posted by mirek on Thu, 14 Feb 2013 19:08:35 GMT
View Forum Message <> Reply to Message

Good idea. Will happen...

Subject: Re: Gcc compile option proposal
Posted by dolik.rce on Thu, 14 Feb 2013 19:46:54 GMT
View Forum Message <> Reply to Message

IMHO it would be much more helpful to use "-Wall", but for that we would have to fix tons of warnings inside U++... I am using Clang and even with the default settings for warnings it outputs so many that theide nearly halts just from parsing all the output Clang produces

OTOH Clang could be used to fix the warnings, as it produces hints how to fix each warning and also has option to produce this output in machine readable form, which could be used to automatically correct the sources (with human review afterwards, of course).

Best regards,
Honza

Subject: Re: Gcc compile option proposal
Posted by mirek on Thu, 14 Feb 2013 19:48:48 GMT

View Forum Message <> Reply to Message

dolik.rce wrote on Thu, 14 February 2013 14:46IMHO it would be much more helpful to use "-Wall", but for that we would have to fix tons of warnings inside U++...

...and inside 3rd party sources...

Subject: Re: Gcc compile option proposal
Posted by Didier on Thu, 14 Feb 2013 20:10:44 GMT

View Forum Message <> Reply to Message

Quote:but for that we would have to fix tons of warnings inside U++
Yes I am perfectly aware of that

Quote: theide nearly halts just from parsing all the output
That's why -Wall may not be such a good idea.

Subject: Re: Gcc compile option proposal
Posted by dolik.rce on Thu, 14 Feb 2013 20:52:25 GMT

View Forum Message <> Reply to Message

Didier wrote on Thu, 14 February 2013 21:10Quote: theide nearly halts just from parsing all the output
That's why -Wall may not be such a good idea.This happens to me on default setting WITHOUT -Wall...

Quote:...and inside 3rd party sources...
Even fixing only the U++ sources would be great improvement. I don't really care about how many warnings is in sources of plugin/* packages. What drives me mad is the warnings that come from the headers that I include in my package, because they are making it harder to notice warnings regarding my code. I believe (but I might be wrong) that there will not be many warnings in the 3rd party headers, at least compared to U++ headers (because there is so many of them in single compilation unit, esp. with BLITZ).

I am well aware that this discussion happened already many times on this forum and that nothing will probably happen this time as ussualy... but I have to at least try to convince you

Honza

Subject: Re: Gcc compile option proposal
Posted by mirek on Thu, 14 Feb 2013 22:01:31 GMT

View Forum Message <> Reply to Message

```
/home/cxl/upp.src/uppsrc/Core/CharSet.h:119:54: warning: '&&' within '||'
[-Wlogical-op-parentheses]
inline bool IsAlpha(int c)        { return c >= 'A' && c <= 'Z' || c >= 'a' && c <= 'z'; }
                                  ~~~~~~~~~~^~~~~~~~~~~ ~~
/home/cxl/upp.src/uppsrc/Core/CharSet.h:119:54: note: place parentheses around the '&&'
expression to silence this warning
```

Go to hell with this... I am not going to make the code unreadable.

Plus, I have a very intense feeling that even if I would screw the code and added parenthesis in this situation, one or two years later somebody would get a nice idea that there are more needed around c >= 'A' etc...

Mirek

Subject: Re: Gcc compile option proposal
Posted by mirek on Thu, 14 Feb 2013 22:21:07 GMT
View Forum Message <> Reply to Message

dolik.rce wrote on Thu, 14 February 2013 15:52
I am well aware that this discussion happened already many times on this forum and that nothing will probably happen this time as ussualy... but I have to at least try to convince you

Honza

It is not true that nothing happened in the past. A couple of years ago, U++ was made GCC/-Wall (mostly) free. Since then -Wall was extended...

(I have also refused to fix 3rd party sources).

It is also BTW impossible to activate full MSC warnings, because they demand "safe" replacement for standard C library functions like strcpy, but those replacements are MSC specific.

Mirek

Subject: Re: Gcc compile option proposal
Posted by mirek on Thu, 14 Feb 2013 22:31:33 GMT
View Forum Message <> Reply to Message

Hm, now I have checked and I am actually compiling with

-Wall -Wno-parentheses

so except that (&&) warning, which I consider really way too stupid, we are -Wall compliant.

Subject: Re: Gcc compile option proposal
Posted by dolik.rce on Fri, 15 Feb 2013 06:41:55 GMT
View Forum Message <> Reply to Message

mirek wrote on Thu, 14 February 2013 23:31Hm, now I have checked and I am actually compiling with

-Wall -Wno-parentheses

so except that (&&) warning, which I consider really way too stupid, we are -Wall compliant.
OK, I could live with blacklisting the parentheses warnings. Other than that, my gcc 4.7.2 reports only one error in theide:Quote:

With clang, it is bit worse:

```
$ umk uppsrc AllForI18n CLANG -ba | grep 'warning:' |  sort -u | grep -o '\[-W.*\' | sort | uniq -c | sort -rn"
     88 [-Wlogical-op-parentheses] *
     40 [-Wdangling-else] *
     16 [-Woverloaded-virtual]
      3 [-Wunused-private-field]
      1 [-Wbitwise-op-parentheses] *
      1 [-Wbool-conversion] *
      1 [-Winvalid-source-encoding] *
      1 [-Wmismatched-tags] *
      1 [-Wtautological-compare]
      1 [-Wunneeded-internal-declaration]
```
Items marked with * are present by default, even without -Wall option. Numbers represents unique occurrences, the actual counts of the errors in output is even higher.

At least some of them should be fixed. I'll gladly volunteer

Honza

Subject: Re: Gcc compile option proposal
Posted by mirek on Fri, 15 Feb 2013 07:33:55 GMT
View Forum Message <> Reply to Message

dolik.rce wrote on Fri, 15 February 2013 01:41mirek wrote on Thu, 14 February 2013 23:31Hm, now I have checked and I am actually compiling with

-Wall -Wno-parentheses

so except that (&&) warning, which I consider really way too stupid, we are -Wall compliant.
OK, I could live with blacklisting the parentheses warnings. Other than that, my gcc 4.7.2 reports only one error in theide:Quote:

With clang, it is bit worse:

Which IMO nicely demonstrates my point. We can "fix" them, but are we going to forever keep changing standard compliant and fairly tested code only because some compiler developer has a little bit different taste for what deserves warning?

(That said, I will go through CLANG warnings anyway to see if there is something I am willing to fix...)

Subject: Re: Gcc compile option proposal
Posted by unodgs on Fri, 15 Feb 2013 10:11:00 GMT
View Forum Message <> Reply to Message

mirek wrote on Fri, 15 February 2013 02:33
Which IMO nicely demonstrates my point. We can "fix" them, but are we going to forever keep changing standard compliant and fairly tested code only because some compiler developer has a little bit different taste for what deserves warning?

Some warnings are simply stupid and this one particularly (one is not using parenthesis because operator precedence rule exists). It should rather be moved to "--best-practices" compiler switch if it existed

---

Subject: Re: Gcc compile option proposal
Posted by Didier on Fri, 15 Feb 2013 16:50:17 GMT
View Forum Message <> Reply to Message

Hi all,

I didn't expect so much passion over such a simple post

But my fisrt post was more about converting a warning into an error rather purly warnings

---

Subject: Re: Gcc compile option proposal
Posted by mirek on Fri, 15 Feb 2013 18:25:14 GMT
View Forum Message <> Reply to Message

I have tried to fix warnings for clang 3.0 (which is default package version for my distro), except that one terror and 3rd party, so for me

clang -Weverything -Wno-parentheses

now compiles without warnings.

Subject: Re: Gcc compile option proposal
Posted by mirek on Sat, 16 Feb 2013 10:16:32 GMT
View Forum Message <> Reply to Message

dolik.rce wrote on Thu, 14 February 2013 14:46IMHO it would be much more helpful to use "-Wall", but for that we would have to fix tons of warnings inside U++... I am using Clang and even with the default settings for warnings it outputs so many that theide nearly halts just from parsing all the output Clang produces

Well, one positive outcome is that I have investigated this choking and optimized it out

Mirek

Subject: Re: Gcc compile option proposal
Posted by nlneilson on Sat, 16 Feb 2013 16:29:02 GMT
View Forum Message <> Reply to Message

Didier wrote on Fri, 15 February 2013 08:50But my fisrt post was more about converting a warning into an error rather purly warnings

If a function should return a value and does not it should throw an error. A default value in a users code if the function for some reason does not come up with one.

I don't understand other options in this thread but it makes my head hurt.

Subject: Re: Gcc compile option proposal
Posted by mirek on Sat, 16 Feb 2013 17:28:53 GMT
View Forum Message <> Reply to Message

Didier wrote on Fri, 15 February 2013 11:50Hi all, I didn't expect so much passion over such a simple post

But my fisrt post was more about converting a warning into an error rather purly warnings

Well, in fact, it is rather configuration issue, you can configure your build method now. (But I plan to make such config default..)

Subject: Re: Gcc compile option proposal
Posted by dolik.rce on Sun, 17 Feb 2013 16:50:59 GMT
View Forum Message <> Reply to Message

mirek wrote on Sat, 16 February 2013 11:16dolik.rce wrote on Thu, 14 February 2013 14:46IMHO it would be much more helpful to use "-Wall", but for that we would have to fix tons of warnings inside U++... I am using Clang and even with the default settings for warnings it outputs so many that theide nearly halts just from parsing all the output Clang produces

Well, one positive outcome is that I have investigated this choking and optimized it out

Mirek

Great, thank you. It is really appreciated...

mirek wrote on Sat, 16 February 2013 11:16dolik.rce wrote on Thu, 14 February 2013 14:46IMHO it would be much more helpful to use "-Wall", but for that we would have to fix tons of warnings inside U++... I am using Clang and even with the default settings for warnings it outputs so many that theide nearly halts just from parsing all the output Clang produces

Well, one positive outcome is that I have investigated this choking and optimized it out

Mirek

... as well as this one.

Best regards,
Honza