

# Re: Program crashes when running it outside dev environment

---

*Source:* [http://coding.derkeiler.com/Archive/C\\_CPP/comp.lang.c/2007-06/msg01370.html](http://coding.derkeiler.com/Archive/C_CPP/comp.lang.c/2007-06/msg01370.html)

---

- *From:* Richard Heathfield <[rjh@xxxxxxxxxxxxxxxxxxx](mailto:rjh@xxxxxxxxxxxxxxxxxxx)>
  - *Date:* Fri, 08 Jun 2007 23:20:25 +0000
- 

Harald van D?k said:

Richard Heathfield wrote:

David Tiktin said:

On 08 Jun 2007, Richard Heathfield  
<[rjh@xxxxxxxxxxxxxxxxxxx](mailto:rjh@xxxxxxxxxxxxxxxxxxx)> wrote:

So my suggestions would be:

- 1) crank up your warning level to the max
- 2) turn off any extensions you can live without
- 3) fix every single diagnostic message, and *\*never\** with a cast

I'm with you right up to the last clause. Are you saying you never actually *\*need\** a cast?

No, there are (rare) occasions when you do need a cast. But I don't know of any occasion where you *\*need\** a cast AND omitting it violates a constraint or constitutes a syntax error. So adding a cast is not the right way to fix a diagnostic message. If the choice is between adding bad code and putting up with a bad warning, I'll live with the bad warning.

I can't think of a concrete example of a cast which causes a constraint violation or syntax error when left out either, but diagnostics are permitted for other code as well.

## Re: Program crashes when running it outside dev environment

Yes indeed, but casts are not the right way to fix these diagnostic messages. (Peace! I know what you're saying.) After all, any compiler can issue any diagnostic message for any reason it likes. For example:

```
int main(void)
{
int i = 42;
return 0;
}
```

Warning: today is Saturday. (Suppress this warning by casting 42 to double.)

Would you obey that suggestion? I certainly wouldn't. (In fact, I'd be tempted to Get A Better Compiler, but of course that isn't always an option.)

Nevertheless, your example still needs to be addressed, so let's read on.

```
#include <stdio.h>
int main(void) {
int *p = NULL;
printf("%p\n", p);
}
```

```
example.c:4: warning: format '%p' expects type 'void *', but argument
2 has type 'int *'
```

And a cast is the right way to fix the bug.

Yes, a cast is the right way to fix the *\*bug\**. The code is broken because a needed cast is absent. But you'd have to fix that *\*anyway\**. The fact that the compiler was kind enough to produce a diagnostic message is neither here nor there.

This may seem like a fine line I'm treading (and perhaps it *\*is\** a fine line), but I think there's an important distinction between "fixing the code" and "changing the code in such a way that you don't get any warnings", especially when implementations have such licence to produce any messages they like. In your example, the cast will actually perform both functions – it will fix the code /and/ suppress the warning. The former is important. The latter, however, is not particularly important.

I used to be one of those who insist on a clean compile. Nowadays, however, I prefer to be one of those who insist on correct code. If I get a clean compile *\*as well\**, that's a bonus. Therefore, when I get

Re: Program crashes when running it outside dev environment

diagnostic messages, I investigate them. If they have any merit, I fix the code. If not, I will probably annotate the code with a comment that explains what warning I'm getting for it, and why, and why I'm not going to change the code. I make an exception for obvious stuff, such as gcc's constant diatribes against code such as `struct foo bar = {0};` which doesn't really merit a comment, as it's so obviously right.

--

Richard Heathfield

"Usenet is a strange place" – dmr 29/7/1999

<http://www.cpax.org.uk>

email: rjh at the above domain, – www.

.