

Re: The annotated annotated annotated C standard

Source: <http://coding.derkeiler.com/Archive/General/comp.programming/2008-01/msg01002.html>

- *From:* Richard Heathfield <rjh@xxxxxxxxxxxxxxxxxx>
 - *Date:* Sun, 20 Jan 2008 07:16:42 +0000
-

spinoza1111 said:

On Jan 20, 6:35 am, "Stephen Howe" <sjhoweATdialDOTpipexDOTcom> wrote:

<snip>

Most of the places where C (and C++ for that matter) standards calls something "undefined" is because the hardware (or OS) may do some unexpected. Like terminate your program.

For example "dividing by 0" is undefined. free()'ing the same pointer twice is undefined.

The standards "undefined" is equivalent to posting a notice on a pond frozen with ice, "THIN ICE".

As long as programmers take care to never allow their programs to skate in that area, their programs will behave well.

If you decide to skate over the "THIN ICE", there is no guarantee what will happen to your program.

This was the case in out of date languages. It is no longer the case.

In fact, a runtime error produced by a bounds checker is preferable to "undefined behavior", as is INF or a similar symbol.

Whether this is preferable depends very much on the situation. What Stephen didn't mention is that there are times when a programmer needs to step outside the Standard in order to achieve a platform-specific goal. When we do this, it very often involves constructs that the Standard declines to define. For example, we might need to point to a particular hardware address that we know exists and is writable on our particular platform. The C Standard can't possibly be expected to know every nuance of every architecture past, present, and future, so it can't tell us what's going to happen when we use such techniques. That is, the behaviour is undefined. Nevertheless, if we know what we're doing, we can get the machine to do what we want, by using our platform-specific knowledge and recognising that we are rendering our code non-portable to other systems.

Re: The annotated annotated annotated C standard

Doug Gwyn once said that q