C11 (C standard revision)

From Wikipedia, the free encyclopedia

C11 (formerly **C1X**) is an informal name for ***ISO/IEC 9899:2011***,[1] the current standard for the C programming language. It replaces the previous C standard, informally known as C99. This new version mainly standardizes features that have already been supported by common contemporary compilers, and includes a detailed memory model to better support multiple threads of execution. Due to delayed availability of conforming C99 implementations, C11 makes certain features optional, to make it easier to comply with the core language standard.[2][3]

The final draft, N1570,[4] was published in April 2011. The new standard passed its final draft review on October 10, 2011 and was officially ratified by ISO and published as ISO/IEC 9899:2011 on December 8, 2011, with no comments requiring resolution by participating national bodies.

A standard macro `__STDC_VERSION__` is defined with value `201112L` to indicate that C11 support is available.[5] Some features of C11 are supported by the GCC starting with version 4.6,[6] Clang starting with version 3.1,[7] and IBM XL C starting with version 12.1.[8]

Contents

Changes from C99

The standard includes several changes to the C99 language and library specifications, such as:[9]

- Alignment specification (`_Alignas` specifier, `_Alignof` operator, `aligned_alloc` function, `<stdalign.h>` header file)
- The `_Noreturn` function specifier and the `<stdnoreturn.h>` header file
- Type-generic expressions using the `_Generic` keyword. For example, the following macro `cbrt(x)` translates to `cbrtl(x)`, `cbrt(x)` or `cbrtf(x)` depending on the type of `x`:

```
#define cbrt(x) _Generic((x), long double: cbrtl, \
                              default: cbrt, \
                              float: cbrtf)(x)
```

- Multi-threading support (`_Thread_local` storage-class specifier, `<threads.h>` header including thread creation/management functions, mutex, condition variable and thread-specific storage functionality, as well as the `_Atomic` type qualifier and `<stdatomic.h>` for uninterruptible object access).

- Improved Unicode support based on the C Unicode Technical Report ISO/IEC TR 19769:2004 (`char16_t` and `char32_t` types for storing UTF-16/UTF-32 encoded data, including conversion functions in `<uchar.h>` and the corresponding `u` and `U` string literal prefixes, as well as the `u8` prefix for UTF-8 encoded literals).[10]
- Removal of the `gets` function, deprecated in the previous C language standard revision, ISO/IEC 9899:1999/Cor.3:2007(E), in favor of a new safe alternative, `gets_s`.
- Bounds-checking interfaces (Annex K).[11]
- Analyzability features (Annex L).
- More macros for querying the characteristics of floating point types, concerning subnormal floating point numbers and the number of decimal digits the type is able to store.
- Anonymous *structures* and *unions*, useful when unions and structures are nested, e.g. in `struct T { int tag; union { float x; int n; }; };`.
- Static assertions, which are evaluated during translation at a later phase than `#if` and `#error`, when types are understood by the translator.
- An exclusive create-and-open mode (`"…x"` suffix) for `fopen`. This behaves like `O_CREAT|O_EXCL` in POSIX, which is commonly used for lock files.
- The `quick_exit` function as a third way to terminate a program, intended to do at least minimal deinitialization if termination with `exit` fails.[12]
- Macros for the construction of complex values (partly because `real + imaginary*I` might not yield the expected value if `imaginary` is infinite or NaN).[13]

Optional features

The new revision allows implementations to not support certain parts of the standard — including some that had been mandatory to support in the 1999 revision.[14] Programs can use predefined macros to determine whether an implementation supports a certain feature or not.

Optional features in C11

Feature	Feature test macro	Availability in C99[15]
Analyzability (Annex L)	`__STDC_ANALYZABLE__`	Not available
Bounds-checking interfaces (Annex K)	`__STDC_LIB_EXT1__`	Not available
Multithreading (`<threads.h>`)	`__STDC_NO_THREADS__`	Not available
Atomic primitives and types (`<stdatomic.h>` and the `_Atomic` type qualifier)[16]	`__STDC_NO_ATOMICS__`	Not available
IEC 60559 floating-point arithmetic (Annex F)	`__STDC_IEC_559__`	Optional
IEC 60559 compatible complex arithmetic (Annex G)	`__STDC_IEC_559_COMPLEX__`	Optional
Complex types (`<complex.h>`)	`__STDC_NO_COMPLEX__`	Mandatory for hosted implementations
Variable length arrays[17]	`__STDC_NO_VLA__`	Mandatory

Criticism

The optional bounds-checking interfaces (Annex K) remain controversial and have not been widely implemented, and their deprecation or removal from the next standard revision has been proposed.[18] (The open-source Open Watcom C/C++ contains a "Safer C" library that is considered a nearly conforming implementation.[19])

See also

- C99, ANSI C, previous standards for the C programming language
- C++17, C++14, C++11, C++03, C++98, versions of the C++ programming language standard
- Compatibility of C and C++

References

1. ISO/IEC 9899:2011 - Information technology - Programming languages - C (http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=57853)
2. WG14 N1250 The C1X Charter (http://www.open-std.org/JTC1/SC22/wg14/www/docs/n1250.pdf)
3. WG14 N1460 Subsetting the C Standard (http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1460.htm)
4. WG14 N1570 Committee Draft — April 12, 2011 (http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1570.pdf)
5. "Defect report #411". ISO/IEC JTC1/SC22/WG14 - C. February 2012. Retrieved 2012-05-04.
6. GCC 4.6 Release Series — Changes, New Features, and Fixes - GNU Project - Free Software Foundation (FSF) (https://gcc.gnu.org/gcc-4.6/changes.html#c)
7. Clang 3.1 Release Notes (http://llvm.org/releases/3.1/docs/ClangReleaseNotes.html#cchanges)
8. Support for ISO C11 added to IBM XL C/C++ compilers (http://www.ibm.com/developerworks/rational/library/support-iso-c11/index.html)
9. WG14 N1516 Committee Draft — October 4, 2010 (http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1516.pdf)
10. WG14 N1286 — "On Support for TR-19769 and New Character Types", Nick Stoughton, Larry Dwyer (http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1286.pdf)
11. Berin Babcock-McConnell. "API02-C. Functions that read or write to or from an array should take an argument to specify the source or target size".
12. WG14 N1327 Abandoning a Process (http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1327.htm)
13. WG14 N1464 Creation of complex value (http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1464.htm)
14. WG14 N1548 Committee Draft — December 2, 2010 (http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1548.pdf) 6.10.8.3 *Conditional feature macros*
15. ISO 9899:1999 6.10.8 *Predefined macro names*
16. WG14 N1558 Mar 14-18 meeting minutes (draft) (http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1558.pdf)
17. ISO 9899:2011 Programming Languages - C 6.7.6.2 4
18. WG14 N1969 — "Updated Field Experience With Annex K — Bounds Checking Interfaces", Carlos O'Donell, Martin Sebor (http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1969.htm)
19. Open Watcom Safer C Library (https://web.archive.org/web/20150503192244/http://openwatcom.org/index.php/Safer_C_Library)

External links

- The C1X Charter (http://www.open-std.org/JTC1/SC22/wg14/www/docs/n1250.pdf)
- N1570 (http://www.open-std.org/jtc1/sc22/wg14/www/docs/n1570.pdf), the final draft of C1X, dated 12 April 2011
- ISO C Working Group's official website (http://www.open-std.org/jtc1/sc22/wg14/)
- Safe C Library of Bounded APIs (http://sourceforge.net/projects/safeclib/)
- Plum, Thomas (April 6, 2012). "C Finally Gets A New Standard". *Dr. Dobb's Journal*.
- Safe C API—Concise solution of buffer overflow, The OWASP Foundation, OWASP AppSec, Beijing 2011 (https://web.archive.org/web/20131203031224/http://www.owasp.org.cn/OWASP_Conference/2011/17_.pdf)

Retrieved from "https://en.wikipedia.org/w/index.php?title=C11_(C_standard_revision)&oldid=749030483"

Categories: C (programming language) | Programming language standards

- This page was last modified on 11 November 2016, at 22:45.
- Text is available under the Creative Commons Attribution-ShareAlike License; additional terms may apply. By using this site, you agree to the Terms of Use and Privacy Policy. Wikipedia® is a registered trademark of the Wikimedia Foundation, Inc., a non-profit organization.