C11

C11标准是 ISO/IEC 9899:2011 - Information technology -- Programming languages -- C 的简称 ，曾用名为C1X。

C11标准是C语言标准的第三版，前一个标准版本是[C99](https://baike.so.com/doc/1583110-1673494.html" \t "https://baike.so.com/doc/_blank)标准。2011年12月8日，[国际标准化组织](https://baike.so.com/doc/5375307-5611383.html" \t "https://baike.so.com/doc/_blank)(ISO)和[国际电工委员会](https://baike.so.com/doc/6509299-6723022.html" \t "https://baike.so.com/doc/_blank)(IEC) 旗下的C语言标准委员会(ISO/IEC JTC1/SC22/WG14)正式发布了C11标准 。

C11标准的最终定稿的草案是免费开放的，为N1570 ，但是正式标准文件需要198瑞士法郎 。

当前，支持此标准的主流C++语言编译器有:[GCC](https://baike.so.com/doc/5725732-5938469.html" \t "https://baike.so.com/doc/_blank)、[Clang](https://baike.so.com/doc/2336404-2471039.html" \t "https://baike.so.com/doc/_blank)、Intel C++ Compiler等。

## 基本信息

中文名称: C11

曾用名: C1X

标准编号: ISO/IEC 9899:2011

发布时间: 2011年12月8日

发布机构: ISO/IEC JTC1/SC22/WG14

## [折叠](https://baike.so.com/doc/7098392-7321344.html)相比C99的变化

1. 对齐处理操作符 alignof，函数 aligned\_alloc()，以及 头文件 <stdalign.h>。见 7.15 节。

2. \_Noreturn 函数标记，类似于 [gcc](https://baike.so.com/doc/5725732-5938469.html" \t "https://baike.so.com/doc/_blank) 的 \_\_attribute\_\_((noreturn))。例子:

\_Noreturn void thrd\_exit(int res);

3. \_Generic 关键词，有点儿类似于 gcc 的 typeof。示例代码:

4. 静态断言( static assertions)，\_Static\_assert()，在解释 #if 和 #error 之后被处理。例子:

\_Static\_assert(FOO > 0, "FOO has a wrong value");

5. 删除了 gets() 函数，C99中已经将此函数被标记为过时，推荐新的替代函数 gets\_s()。

6. 新的 fopen() 模式，("…x")。类似 POSIX 中的 O\_CREAT|O\_EXCL，在文件锁中比较常用。

7. 匿名结构体/联合体，这个早已经在 gcc 中了，我们并不陌生，定义在 6.7.2.1 p13。

8. 多线程支持，包括:\_Thread\_local，头文件 <threads.h>，里面包含线程的创建和管理函数(比如 thrd\_create()，thrd\_exit())，mutex (比如 mtx\_lock()，mtx\_unlock())等等，更多内容清参考 7.26 节。

9. \_Atomic类型修饰符和 头文件 <stdatomic.h>，见 7.17 节。

10. 带边界检查(Bounds-checking)的函数接口，定义了新的安全的函数，例如 fopen\_s()，strcat\_s() 等等。更多参考 Annex K。

11. 改进的 [Unicode](https://baike.so.com/doc/4443890-4652181.html" \t "https://baike.so.com/doc/_blank) 支持，新的头文件 <uchar.h> 等。实例代码:

12. 新增 quick\_exit() 函数，作为第三种终止程序的方式，当 exit() 失败时可以做最少的清理工作(deinitializition)，具体见 7.22.4.7。

13. 创建复数的宏， CMPLX()，见 7.3.9.3。

14. 更多浮点数处理的宏 (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)。

15. struct timespec 成为 time.h 的一部分，以及宏 TIME\_UTC，函数 timespec\_get()。

## C11还缺少什么

1、对IEEE754标准-2008的半精度浮点数的支持。尽管遵循GNU C标准规范的编译器([GCC](https://baike.so.com/doc/5725732-5938469.html" \t "https://baike.so.com/doc/_blank)以及[Clang](https://baike.so.com/doc/2336404-2471039.html" \t "https://baike.so.com/doc/_blank)等)使用了C扩展关键字\_\_fp16，但是只能对半精度浮点数进行引用，而不能做任何算术操作。因此，这个特性应当在下一个[C语言](https://baike.so.com/doc/5347427-5582874.html" \t "https://baike.so.com/doc/_blank)标准中立马加入进去的，建议使用\_Half作为半精度浮点数类型的关键字。

2、匿名函数:匿名函数，也叫lambda表达式，是现代[计算机编程语言](https://baike.so.com/doc/6222896-6436204.html" \t "https://baike.so.com/doc/_blank)的一个典范特性，在Clang编译器中已经通过blocks语法 进行了支持，该语法也被发表到了下一代C标准的提案 。Lambda表达式非常适用于多核多线程[并行计算](https://baike.so.com/doc/6555550-6769301.html" \t "https://baike.so.com/doc/_blank)，而不仅仅只是用于[语法糖](https://baike.so.com/doc/1355052-1432578.html" \t "https://baike.so.com/doc/_blank)。

3、增加对[UTF-16](https://baike.so.com/doc/7060256-7283166.html" \t "https://baike.so.com/doc/_blank)字符编码字符串格式符的支持。在标准[C语言](https://baike.so.com/doc/5347427-5582874.html" \t "https://baike.so.com/doc/_blank)中，直到C11还只能支持[UTF-8](https://baike.so.com/doc/7105043-7328054.html" \t "https://baike.so.com/doc/_blank)编码字符串的格式，采用%s。而对于UTF-16编码字符串的各类操作都比较欠缺，而只有UTF-16转UTF-8的库函数c16rtomb以及UTF-8转UTF-16的标准库函数mbrtoc16可使用，因此在实际项目工程上，用起来十分繁琐。这里建议后续标准C语言采用%S表示UTF-16的字符串格式符，%s作为UTF-8字符串格式符，%C表示UTF-16字符，%c表示UTF-8字符。