**jobbole/awesome-cpp-cn**

[@tangyouhua](https://github.com/tangyouhua)

[**tangyouhua**](https://github.com/jobbole/awesome-cpp-cn/commits?author=tangyouhua) [Merge pull request](https://github.com/jobbole/awesome-cpp-cn/commit/8cce7ab122386b61b3e901d318b5b75c9737a2ff) [#23](https://github.com/jobbole/awesome-cpp-cn/pull/23) [from ShenMian/master](https://github.com/jobbole/awesome-cpp-cn/commit/8cce7ab122386b61b3e901d318b5b75c9737a2ff)

…

**Files**

Type

Name

Latest commit message

Commit time

[README.md](https://github.com/jobbole/awesome-cpp-cn/blob/master/README.md)

[格式化](https://github.com/jobbole/awesome-cpp-cn/commit/63a64513fdf93fcb5977c82581204dab1dc87ea9)

8 months ago

[**README.md**](https://github.com/jobbole/awesome-cpp-cn#readme)

**C++ 资源大全中文版**

[awesome-cpp](https://github.com/fffaraz/awesome-cpp) 就是 fffaraz 发起维护的 C++ 资源列表，内容包括: 标准库、Web应用框架、人工智能、数据库、图片处理、机器学习、日志、代码分析等。

中文版由开源前哨和CPP开发者微信公号团队维护更新，在 GitHub 已有近4400 Star，欢迎在 Github 上关注。

**本项目的参与者**

* 维护者: 开源前哨和CPP开发者微信公号团队。 「开源前哨」会定期在知乎专栏分享最新、有趣和热门的开源项目，每个项目都有详细的介绍和示例。传送门: <https://www.zhihu.com/column/c_1317124962785062912>
* 贡献者: [cccookieee](https://github.com/cccookieee)、JingerJoe、云中游、冰斌、Juliesand、[ZZMarquis](https://github.com/ZZMarquis)、[颜闽辉](https://github.com/yanminhui/)、[tangyouhua](https://github.com/tangyouhua)、[ShenMian](https://github.com/ShenMian)

注: 名单不分排名，不定期补充更新

**目录**

* [标准库](https://github.com/jobbole/awesome-cpp-cn#%E6%A0%87%E5%87%86%E5%BA%93)
* [框架](https://github.com/jobbole/awesome-cpp-cn#%E6%A1%86%E6%9E%B6)
* [人工智能](https://github.com/jobbole/awesome-cpp-cn#%E4%BA%BA%E5%B7%A5%E6%99%BA%E8%83%BD)
* [异步事件循环](https://github.com/jobbole/awesome-cpp-cn#%E5%BC%82%E6%AD%A5%E4%BA%8B%E4%BB%B6%E5%BE%AA%E7%8E%AF)
* [音频](https://github.com/jobbole/awesome-cpp-cn#%E9%9F%B3%E9%A2%91)
* [生态学](https://github.com/jobbole/awesome-cpp-cn#%E7%94%9F%E6%80%81%E5%AD%A6)
* [压缩](https://github.com/jobbole/awesome-cpp-cn#%E5%8E%8B%E7%BC%A9)
* [并发性](https://github.com/jobbole/awesome-cpp-cn#%E5%B9%B6%E5%8F%91%E6%80%A7)
* [容器](https://github.com/jobbole/awesome-cpp-cn#%E5%AE%B9%E5%99%A8)
* [密码学](https://github.com/jobbole/awesome-cpp-cn#%E5%AF%86%E7%A0%81%E5%AD%A6)
* [数据库](https://github.com/jobbole/awesome-cpp-cn#%E6%95%B0%E6%8D%AE%E5%BA%93)
* [调试](https://github.com/jobbole/awesome-cpp-cn#%E8%B0%83%E8%AF%95)
* [游戏引擎](https://github.com/jobbole/awesome-cpp-cn#%E6%B8%B8%E6%88%8F%E5%BC%95%E6%93%8E)
* [图形用户界面](https://github.com/jobbole/awesome-cpp-cn#%E5%9B%BE%E5%BD%A2%E7%94%A8%E6%88%B7%E7%95%8C%E9%9D%A2)
* [图形](https://github.com/jobbole/awesome-cpp-cn#%E5%9B%BE%E5%BD%A2)
* [图像处理](https://github.com/jobbole/awesome-cpp-cn#%E5%9B%BE%E5%83%8F%E5%A4%84%E7%90%86)
* [国际化](https://github.com/jobbole/awesome-cpp-cn#%E5%9B%BD%E9%99%85%E5%8C%96)
* [Json](https://github.com/jobbole/awesome-cpp-cn#Json)
* [日志](https://github.com/jobbole/awesome-cpp-cn#%E6%97%A5%E5%BF%97)
* [机器学习](https://github.com/jobbole/awesome-cpp-cn#%E6%9C%BA%E5%99%A8%E5%AD%A6%E4%B9%A0)
* [数学](https://github.com/jobbole/awesome-cpp-cn#%E6%95%B0%E5%AD%A6)
* [多媒体](https://github.com/jobbole/awesome-cpp-cn#%E5%A4%9A%E5%AA%92%E4%BD%93)
* [网络](https://github.com/jobbole/awesome-cpp-cn#%E7%BD%91%E7%BB%9C)
* [物理学](https://github.com/jobbole/awesome-cpp-cn#%E7%89%A9%E7%90%86%E5%AD%A6)
* [机器人学](https://github.com/jobbole/awesome-cpp-cn#%E6%9C%BA%E5%99%A8%E4%BA%BA%E5%AD%A6)
* [科学计算](https://github.com/jobbole/awesome-cpp-cn#%E7%A7%91%E5%AD%A6%E8%AE%A1%E7%AE%97)
* [脚本](https://github.com/jobbole/awesome-cpp-cn#%E8%84%9A%E6%9C%AC)
* [序列化](https://github.com/jobbole/awesome-cpp-cn#%E5%BA%8F%E5%88%97%E5%8C%96)
* [视频](https://github.com/jobbole/awesome-cpp-cn#%E8%A7%86%E9%A2%91)
* [虚拟机](https://github.com/jobbole/awesome-cpp-cn#%E8%99%9A%E6%8B%9F%E6%9C%BA)
* [Web应用框架](https://github.com/jobbole/awesome-cpp-cn#Web%E5%BA%94%E7%94%A8%E6%A1%86%E6%9E%B6)
* [XML](https://github.com/jobbole/awesome-cpp-cn#XML)
* [多项混杂](https://github.com/jobbole/awesome-cpp-cn#%E5%A4%9A%E9%A1%B9%E6%B7%B7%E6%9D%82)
* [软件](https://github.com/jobbole/awesome-cpp-cn#%E8%BD%AF%E4%BB%B6)
* [编译器](https://github.com/jobbole/awesome-cpp-cn#%E7%BC%96%E8%AF%91%E5%99%A8)
* [在线编译器](https://github.com/jobbole/awesome-cpp-cn#%E5%9C%A8%E7%BA%BF%E7%BC%96%E8%AF%91%E5%99%A8)
* [调试器](https://github.com/jobbole/awesome-cpp-cn#%E8%B0%83%E8%AF%95%E5%99%A8)
* [集成开发环境](https://github.com/jobbole/awesome-cpp-cn#%E9%9B%86%E6%88%90%E5%BC%80%E5%8F%91%E7%8E%AF%E5%A2%83)
* [构建系统](https://github.com/jobbole/awesome-cpp-cn#%E6%9E%84%E5%BB%BA%E7%B3%BB%E7%BB%9F)
* [静态代码分析](https://github.com/jobbole/awesome-cpp-cn#%E9%9D%99%E6%80%81%E4%BB%A3%E7%A0%81%E5%88%86%E6%9E%90)

**标准库**

C++标准库，包括了STL容器，算法和函数等。

* [C++ Standard Library](http://en.wikipedia.org/wiki/C%2B%2B_Standard_Library): 是一系列类和函数的集合，使用核心语言编写，也是C++ISO自身标准的一部分。
* [Standard Template Library](http://en.wikipedia.org/wiki/Standard_Template_Library): 标准模板库。
* [C POSIX library](http://en.wikipedia.org/wiki/C_POSIX_library): POSIX系统的C标准库规范。
* [ISO C++ Standards Committee](https://github.com/cplusplus): C++标准委员会。
* [The GNU C Library](https://www.gnu.org/software/libc/manual): 本手册的目的是告诉你如何使用GNU C库。

**框架**

C++通用框架和库

* [abseil-cpp](https://github.com/abseil/abseil-cpp): Abseil C++ 通用库。
* [Apache C++ Standard Library](http://stdcxx.apache.org/): 是一系列算法，容器，迭代器和其他基本组件的集合。
* [APR](http://apr.apache.org/): Apache可移植运行时库。另一个跨平台的库。
* [ASL](http://stlab.adobe.com/): Adobe源代码库提供了同行的评审和可移植的 C++ 源代码库。
* [Boost](https://github.com/boostorg): 大量通用 C++ 库的集合。
* [BDE](https://github.com/bloomberg/bde): 来自于彭博资讯实验室的开发环境。
* [Cinder](http://libcinder.org/): 提供专业品质创造性编码的开源开发社区。
* [CO](https://github.com/idealvin/co) - C++基础库，包含 go-style 协程库，日志库, JSON库, RPC 框架以及其他基本组件。
* [Cxxomfort](http://ryan.gulix.cl/fossil.cgi/cxxomfort/): 轻量级的，只包含头文件的库，将 C++ 11的一些新特性移植到C++03中。
* [Dlib](http://dlib.net/): 使用契约式编程和现代 C++ 科技设计的通用的跨平台的 C++ 库。
* [EASTL](https://github.com/paulhodge/EASTL): EA-STL公共部分。
* [ETL](https://github.com/ETLCPP/etl): 嵌入式模板库。
* [ffead-cpp](https://github.com/sumeetchhetri/ffead-cpp): 企业应用程序开发框架。
* [Folly](https://github.com/facebook/folly): 由Facebook开发和使用的开源C++库。
* [FunctionalPlus](https://github.com/Dobiasd/FunctionalPlus): C++ 的函数式编程库，编写简洁易读的C++代码。
* [GLib](https://wiki.gnome.org/Projects/GLib): GLib提供用C编写的库和应用的核心代码块。
* [JUCE](https://github.com/julianstorer/JUCE): 包罗万象的C++类库，用于开发跨平台软件。
* [Kigs framework](https://github.com/Kigs-framework/kigs): C++免费开源的模块化、多功能、跨平台的RAD框架。
* [libPhenom](https://github.com/facebook/libphenom): 用于构建高性能和高度可扩展性系统的事件框架。
* [LibSourcey](https://github.com/sourcey/libsourcey): 用于实时的视频流和高性能网络应用程序的C++11 evented IO。
* [LibU](https://github.com/koanlogic/libu): C语言写的多平台工具库。
* [Loki](http://loki-lib.sourceforge.net/): C++库的设计，包括常见的设计模式和习语的实现。
* [MiLi](https://code.google.com/p/mili/): 只含头文件的小型C++库。
* [openFrameworks](http://www.openframeworks.cc/): 开发C++工具包，用于创意性编码。
* [Qt](http://qt-project.org/): 跨平台的应用程序和用户界面框架。
* [Reason](http://code.google.com/p/reason/): 跨平台的框架，使开发者能够更容易地使用Java，.Net和Python，同时也满足了他们对C++性能和优势的需求。
* [ROOT](http://root.cern.ch/): 具备所有功能的一系列面向对象的框架，能够非常高效地处理和分析大量的数据，为欧洲原子能研究机构所用。
* [STLport](http://www.stlport.org/): 是STL具有代表性的版本。
* [STXXL](http://stxxl.sourceforge.net/): 用于额外的大型数据集的标准模板库。
* [tbox](https://github.com/tboox/tbox): 类似于glib的多平台C库。
* [Ultimate++](http://www.ultimatepp.org/): C++跨平台快速应用程序开发框架。
* [uSTL](http://msharov.github.io/ustl/): 小型STL库。
* [Windows Template Library](http://sourceforge.net/projects/wtl/): 用于开发Windows应用程序和UI组件的C++库。
* [Yomm2](https://github.com/jll63/yomm2): 快速的、正交的、开放的multi-methods，取代[Yomm11](https://github.com/jll63/yomm11)。

**人工智能**

* [ANNetGPGPU](https://github.com/ANNetGPGPU/ANNetGPGPU): 基于GPU(CUDA)的人工神经网络库。
* [btsk](https://github.com/aigamedev/btsk): 游戏行为树启动器工具。
* [Evolving Objects](http://eodev.sourceforge.net/): 基于模板的，ANSI C++演化计算库，能够帮助你非常快速地编写出自己的随机优化算法。
* [frugally-deep](https://github.com/Dobiasd/frugally-deep): 使用Keras模型的C++库，只有头文件。
* [Genann](https://github.com/codeplea/genann): 简单的神经网络C库。
* [MXNet](https://github.com/apache/incubator-mxnet): 轻量级、便携、灵活的分布式/移动深度学习，具有动态、可感知突变的数据流管理调度程序。适用于Python、R、Julia、Scala、Go、Javascript和更多网站。
* [PyTorch](https://github.com/pytorch/pytorch): 具有强大CPU加速功能，包含张量和动态神经网络的Python库。
* [Recast/Detour](https://github.com/recastnavigation/recastnavigation): （3D）导航网格生成器和路径查找，主要用于游戏。
* [TensorFlow](https://github.com/tensorflow/tensorflow): 使用数据流图进行数值计算的开源软件库。
* [CNTK](https://github.com/Microsoft/CNTK): Microsoft Cognitive Toolkit (CNTK)，开源的深度学习工具库。
* [tiny-dnn](https://github.com/tiny-dnn/tiny-dnn): C++11中的无依赖项的深度学习框架，只有头文件。
* [Veles](https://github.com/Samsung/veles): 用于快速深度学习应用开发的分布式平台。
* [Kaldi](https://github.com/kaldi-asr/kaldi): 语音识别工具包。

**异步事件循环**

* [Asio](https://github.com/chriskohlhoff/asio/): 用于网络和底层I/O编程的跨平台 C++ 库，使用现代 C++ 方法为开发者提供一致的异步模型。
* [Boost.Asio](http://think-async.com/): 用于网络和底层I/O编程的跨平台的 C++ 库。
* [C++ Actor Framework](https://github.com/actor-framework/actor-framework): C++中Actor Model的开源实现。
* [libev](http://libev.schmorp.de/): 功能齐全，高性能的时间循环，轻微地仿效libevent，但是不再像libevent一样有局限性，也修复了它的一些bug。
* [libevent](http://libevent.org/): 事件通知库。
* [libhv](https://github.com/ithewei/libhv): 跨平台的事件循环库。
* [libuv](https://github.com/joyent/libuv): 跨平台异步I/O。
* [promise-cpp](https://github.com/xhawk18/promise-cpp): 实现Promise/A+标准的库，只有头文件。
* [uvw](https://github.com/skypjack/uvw): libuv的C++封装器。

**音频**

音频，声音，音乐，数字化音乐库

* [AudioFile](https://github.com/adamstark/AudioFile): 用于读写音频文件的简单C++库。
* [FMOD](http://www.fmod.org/): 易于使用的跨平台的音频引擎和音频内容的游戏创作工具。
* [KFR](https://www.kfrlib.com/): 快速、现代的C++ DSP框架，FFT、FIR/IIR滤波器，和采样率转换。
* [LAME](https://lame.sourceforge.io/using.php): LAME是高质量MPEG音频第三层（MP3）编码器。
* [libsndfile](https://github.com/erikd/libsndfile/): C++封装的C库，用于通过标准库接口读写包含采样声音的文件。
* [libsoundio](https://github.com/andrewrk/libsoundio): 用于跨平台实时音频输入输出的C库。
* [Maximilian](https://github.com/micknoise/Maximilian): C++音频和音乐数字信号处理库。
* [OpenAL](http://www.openal.org/): 开源音频库---跨平台的音频API。
* [miniaudio](https://github.com/dr-soft/miniaudio): 单文件音频回放和录制的库。
* [Opus](http://opus-codec.org/): 一个完全开放的，免版税的，高度通用的音频编解码器。
* [PortAudio](http://www.portaudio.com/): 免费的、跨平台的、开源的音频I/O库。
* [SELA](https://github.com/sahaRatul/sela): SimplIE无损音频。
* [SoLoud](https://github.com/jarikomppa/soloud): 简单便携的游戏音频引擎。
* [Speex](http://www.speex.org/): 免费编解码器，为Opus所废弃。
* [Tonic](https://github.com/TonicAudio/Tonic): C++易用和高效的音频合成。
* [Vorbis](http://xiph.org/vorbis/): Ogg Vorbis是一种完全开放的，非专有的，免版税的通用压缩音频格式。
* [minimp3](https://github.com/lieff/minimp3): 公共域，无尘室实现的MP3解码器，只有头文件。
* [Verovio](https://github.com/rism-ch/verovio): 快速、轻量级的音乐符号雕刻库。
* [Wav2Letter++](https://github.com/facebookresearch/wav2letter/): 公共域，完全用C++编写的快速开放源代码的语音处理库，同时使用ArrayFire张量库和flashlight机器学习库来实现最高的效率。

**生态学**

生物信息，基因组学和生物技术

* [BioC++](http://biocpp.sourceforge.net/): 生物信息学的C++计算库。
* [Chaste](http://www.cs.ox.ac.uk/chaste/): 开放源代码的C++库，用于为生理学和生物学开发的数学模型的计算仿真。
* [libsequence](http://molpopgen.github.io/libsequence/): 用于表示和分析群体遗传学数据的C++库。
* [SeqAn](http://www.seqan.de/): 专注于生物数据序列分析的算法和数据结构。
* [Vcflib](https://github.com/ekg/vcflib): 用于解析和处理VCF文件的C++库。
* [Wham](https://github.com/jewmanchue/wham): 直接把联想测试应用到BAM文件的基因结构变异。

**比特流**

* [jech/dht](https://github.com/jech/dht): C实现的比特流分布式哈希表库。
* [libtorrent](https://github.com/arvidn/libtorrent)(a.k.a. libtorrent-rasterbar): 高效的、功能完整的C++比特流实现。
* [LibTorrent](https://github.com/rakshasa/libtorrent) (a.k.a. libtorrent-rakshasa): 比特流库。
* [libutp](https://github.com/bittorrent/libutp): uTorrent传输协议库。

**化学**

化学，地球化学，生物化学

* [d-SEAMS](https://github.com/d-SEAMS/seams-core): 具有Nix的C++和Lua实现的分子动力学轨道分析引擎，它是分子模拟的递延结构分析的首字母缩写。
* [gromacs](https://github.com/gromacs/gromacs): 传递消息的并行分子动力学实现。
* [Reaktoro](https://github.com/reaktoro/reaktoro): C++和python中用于对化学反应系统进行建模的计算框架。
* [LAMMPS](https://github.com/lammps/lammps): 经典的用于材料建模的分子动力学代码，它是大范围原子/分子巨量平行模拟器的首字母缩写。

**命令行界面**

控制台/终端用户界面，命令行界面

* [Argh!](https://github.com/adishavit/argh): 极简的只有头文件的参数处理程序。
* [Taywee / args](https://github.com/taywee/args): 简单的只有头文件的C++参数解析库。
* [Boost.Program\_options](http://www.boost.org/doc/libs/1_57_0/doc/html/program_options.html): 通过常见的方法，例如命令行和配置文件，来获取程序选择项的库。
* [Clara](https://github.com/catchorg/Clara): 用于C++11及更高版本的易于使用、可组合的命令行解析器。
* [cli](https://github.com/daniele77/cli): 用于交互式命令行界面的跨平台的C++14库，只有头文件。
* [CLI11](https://github.com/CLIUtils/CLI11): 单文件或多文件的C++11库，用于简单和高级的CLI解析，只有头文件。
* [jarro2783/cxxopts](https://github.com/jarro2783/cxxopts): 轻量级的C++命令行选项解析器。
* [docopt.cpp](https://github.com/docopt/docopt.cpp): 从文档字符串生成选项解析的库。
* [gflags](https://gflags.github.io/gflags/): C++的命令行标志模块。
* [indicators](https://github.com/p-ranav/indicators/): 现代C++的活动指标。
* [linenoise](https://github.com/antirez/linenoise): readline和libedit的独立小选择。
* [linenoise-ng](https://github.com/arangodb/linenoise-ng): 用于Linux、Windows和MacOS的小型便携式GNU readline替换，它能够处理UTF-8字符。
* [Lyra](https://github.com/bfgroup/Lyra): 易于使用的、可组合的命令行解析器，适用于C++11及更高版本。
* [Ncurses](http://invisible-island.net/ncurses/): 终端用户界面。
* [PDCurses](https://github.com/wmcbrine/PDCurses): 具有源代码和预编译库的公共域curses库。
* [replxx](https://github.com/AmokHuginnsson/replxx): 支持UTF-8、语法高亮、提示的readline和libedit替换，可在Unix和Windows上工作。
* [tabulate](https://github.com/p-ranav/tabulate): 现代C++的表格制作工具。
* [TCLAP](http://tclap.sourceforge.net/): 用于在ANSI C++中定义和访问命令行参数的成熟、稳定、功能多样的库。
* [termbox](https://github.com/nsf/termbox): 用于编写基于文本的用户界面的C库。

**压缩**

压缩和归档库

* [bit7z](https://github.com/rikyoz/bit7z): C++静态库，提供与7-zip DLLs的简洁接口。
* [Brotli](https://github.com/google/brotli): Brotli压缩格式，由Google开发。
* [bzip2](http://www.bzip.org/): 一个完全免费，免费专利和高质量的数据压缩。
* [FiniteStateEntropy](https://github.com/Cyan4973/FiniteStateEntropy): 新一代熵编解码器: 有限状态熵和Huff()。
* [PhysicsFS](https://icculus.org/physfs/): 对各种归档提供抽象访问的库，主要用于视频游戏，设计灵感部分来自于Quake3的文件子系统。
* [KArchive](https://projects.kde.org/projects/frameworks/karchive): 用于创建，读写和操作文件档案（例如zip和 tar）的库，它通过QIODevice的一系列子类，使用gzip格式，提供了透明的压缩和解压缩的数据。
* [libarchive](https://github.com/libarchive/libarchive): 多格式的存档和压缩库。
* [LZ4](https://code.google.com/p/lz4/): 非常快速的压缩算法。
* [LZFSE](https://github.com/lzfse/lzfse): LZFSE压缩库和命令行工具。
* [LZHAM](https://code.google.com/p/lzham/): 无损压缩数据库，压缩比率跟LZMA接近，但是解压缩速度却要快得多。
* [LZMA](http://www.7-zip.org/sdk.html): 7z格式默认和通用的压缩方法。
* [LZMAT](http://www.matcode.com/lzmat.htm): 极其快速的实时无损数据压缩库。
* [miniz](https://code.google.com/p/miniz/): 单一的C源文件，紧缩/膨胀压缩库，使用zlib兼容API，ZIP归档读写，PNG写方式。
* [Minizip](https://github.com/nmoinvaz/minizip): Zlib最新bug修复，支持PKWARE磁盘跨越，AES加密和IO缓冲。
* [smaz](https://github.com/antirez/smaz): 小型字符压缩库。
* [Snappy](https://code.google.com/p/snappy/): 快速压缩和解压缩。
* [ZLib](http://zlib.net/): 非常紧凑的数据流压缩库。
* [zlib-ng](https://github.com/Dead2/zlib-ng): 用于“下一代”系统的zlib，将一些重要的优化进行嵌入式替换。
* [zstd](https://github.com/facebook/zstd): Zstandard-快速实时压缩算法。由Facebook开发。
* [ZZIPlib](http://zziplib.sourceforge.net/): 提供ZIP归档的读权限。

**并发性**

并发执行和多线程

* [alpaka](https://github.com/ComputationalRadiationPhysics/alpaka): 并行内核加速的抽象库。
* [ArrayFire](https://github.com/arrayfire/arrayfire): 通用GPU库。
* [Async++](https://github.com/Amanieu/asyncplusplus): C++ 11 的轻量级并行框架，受Microsoft PPL库和N3428 C++标准提案启发。
* [Boost.Compute](https://github.com/kylelutz/compute): 用于OpenCL的C++GPU计算库。
* [Bolt](https://github.com/HSA-Libraries/Bolt): 针对GPU进行优化的C++模板库。
* [ck](https://github.com/concurrencykit/ck): 并发基元，安全内存回收机制和非阻塞数据结构。
* [concurrentqueue](https://github.com/cameron314/concurrentqueue): C++11的快速多生产者、多消费者的无锁并发队列。
* [Cpp-Taskflow](https://github.com/cpp-taskflow/cpp-taskflow): 具有任务依赖性的快速C++并行编程。
* [CUB](https://github.com/NVlabs/cub): CUB为CUDA编程模式的每一层提供了最新的可重用软件组件。
* [cuda-api-wrappers](https://github.com/eyalroz/cuda-api-wrappers): 轻量级的现代C++封装器，用于CUDA GPU的运行时API编程。
* [cupla](https://github.com/ComputationalRadiationPhysics/cupla): 通过Alpaka在OpenMPA、线程、TBB……运行CUDA/C++的C++ API。
* [C++React](https://github.com/schlangster/cpp.react): 用于C++11的反应性编程库。
* [FiberTaskingLib](https://github.com/RichieSams/FiberTaskingLib): 基于任务的多线程库，支持具有任意依赖关系的任务图表。
* [HPX](https://github.com/STEllAR-GROUP/hpx/): 适用于任何规模的并行分布式应用的通用C++运行时系统。
* [Intel Games Task Scheduler](https://github.com/GameTechDev/GTS-GamesTaskScheduler): 为了游戏开发者的需要而设计的任务调度框架。
* [Intel Parallel STL](https://github.com/intel/parallelstl): C++ 17 STL的英特尔实现，适用于C++11及更高版本。
* [Intel TBB](https://www.threadingbuildingblocks.org/): 英特尔线程构建模块。
* [junction](https://github.com/preshing/junction): 并发数据结构的C++库。
* [Kokkos](https://github.com/kokkos/kokkos): 用于并行运行和内存抽象的便携式编程模型。
* [libcds](https://github.com/khizmax/libcds): 并行数据结构的C++库。
* [Libclsph](https://github.com/libclsph/libclsph): 基于OpenCL的GPU加速SPH流体仿真库。
* [libmill](https://github.com/sustrik/libmill/): 在C中引入结构并发性。
* [libdispatch](https://github.com/apple/swift-corelibs-libdispatch): Apple公司开发的Grand Central Dispatch（GCD）是基于线程池模式的任务并行技术。libdispatch库提供了GCD服务的实现。
* [libmill](https://github.com/sustrik/libmill/): 在C中引入Go-style并发性。
* [marl](https://github.com/google/marl): Marl是用C++11编写的混合线程/纤程的任务调度程序。
* [moderngpu](https://github.com/moderngpu/moderngpu): moderngpu是用于GPUs通用计算的生产力库，它只有为CUDA编写的C++头文件。该库的独特价值在于其用于解决不规则并行问题的加速基元。
* [NCCL](https://github.com/NVIDIA/nccl): 用于集体多GPU通信的优化基元。
* [OpenCL](https://www.khronos.org/opencl/): 并行编程的异构系统的开放标准。
* [OpenMP](http://openmp.org/): OpenMP API。
* [SObjectizer](https://github.com/Stiffstream/sobjectizer): 实现Actor、Publish-Subscribe和CSP模式的相当小的C++框架。
* [Quantum](https://github.com/bloomberg/quantum): 建立在[boost::coroutines2](https://www.boost.org/doc/libs/1_65_0/libs/coroutine2/doc/html/index.html)顶层的强大的C++协同调度程序框架。
* [RaftLib](http://raftlib.io/): RaftLib C++ 库，通过类似iostream的 C++ 运算符实现流/数据流并发性。
* [readerwriterqueue](https://github.com/cameron314/readerwriterqueue): C++的快速单生产者、单消费者的无锁队列。
* [stdgpu](https://github.com/stotko/stdgpu): GPU上高效的类似STL的数据结构。
* [Thrust](http://thrust.github.io/): 类似于C++标准模板库的并行算法库。
* [transwarp](https://github.com/bloomen/transwarp): 任务并发性的C++库，只有头文件。
* [VexCL](https://github.com/ddemidov/vexcl): 用于OpenCL/CUDA 的C++向量表达式模板库。
* [Quantum](https://github.com/bloomberg/quantum): 建立在[boost::coroutines2](https://www.boost.org/doc/libs/1_65_0/libs/coroutine2/doc/html/index.html)顶层的强大的C++协同调度程序框架。
* [STAPL](http://parasol-lab.gitlab.io/stapl-home/): C++并行编程框架，旨在工作在共享和分布式内存并行电脑上。

**配置**

配置文件，INI文件

* [inih](https://github.com/benhoyt/inih): C语言的简单.INI文件解析器，很适合嵌入式系统。
* [inih](https://github.com/jtilly/inih): [inih](https://github.com/benhoyt/inih)的C++版本，只有头文件。
* [iniparser](https://github.com/ndevilla/iniparser): INI文件解析器。
* [libconfig](https://github.com/hyperrealm/libconfig): 用于处理结构化配置文件的C、C++库。
* [libconfuse](https://github.com/martinh/libconfuse): C的小型配置文件解析库。
* [simpleini](https://github.com/brofield/simpleini): 跨平台的C++库，提供用于读写INI配置文件的简单API。
* [toml++](https://github.com/marzer/tomlplusplus): TOML解析器和串化器，适用于C++17及更高版本，只有头文件。

**容器**

* [C++ B-tree](https://code.google.com/p/cpp-btree/): 基于B树数据结构，实现命令内存容器的模板库。
* [Colony](https://github.com/mattreecebentley/plf_colony): 无序的”bag“型容器，在高度修改的情况下，其性能优于标准容器。同时，无论插入还是擦除，其都能始终保持指向未擦除元素的永久指针。
* [dynamic\_bitset](https://github.com/pinam45/dynamic_bitset): C++17 的动态位集合，只有头文件。
* [Forest](https://github.com/xorz57/forest): 实现了AVL、二进制搜索、KD和四叉树的模板库。
* [Hashmaps](https://github.com/goossaert/hashmap): C++中开放寻址哈希表算法的实现。
* [Hopscotch map](https://github.com/Tessil/hopscotch-map): 使用hopscotch哈希算法来实现冲突解决的快速哈希映射，只有头文件。
* [LSHBOX](https://github.com/RSIA-LIESMARS-WHU/LSHBOX): 局部敏感算法（LSH）的C++工具箱，提供了好几种普遍的LSH算法，也可以支持Python和MATLAB。
* [PGM-index](https://github.com/gvinciguerra/PGM-index): 能够快速查找、前导、范围搜索和更新数以亿计项数组的数据结构，其跟传统索引相比，使用梯度的顺序而不是空间。
* [plf::list](https://github.com/mattreecebentley/plf_list): std::list实现，其清除范围拼接以启用缓存友好的结构，从而显著提高性能。
* [plf::stack](https://github.com/mattreecebentley/plf_stack): std::stack容器适配器的替换容器，其在堆栈上下文中比任何标准容器都有更好的性能。
* [ring\_span lite](https://github.com/martinmoene/ring-span-lite): Arthur O'Dwyer的ring\_span的简化实现，例如循环缓冲视图。
* [robin-hood-hashing](https://github.com/martinus/robin-hood-hashing): 用于C++1的快速、节约内存的哈希表，基于robin hood哈希算法。
* [robin-map](https://github.com/Tessil/robin-map): 使用robin hood哈希算法的哈希映射和哈希集。
* [sparsepp](https://github.com/greg7mdp/sparsepp): 用于C++的快速、节约内存的哈希映射。

**密码学**

密码学和加密库

* [Bcrypt](http://bcrypt.sourceforge.net/): 一个跨平台的文件加密工具，加密文件可以移植到所有可支持的操作系统和处理器中。
* [BeeCrypt](https://github.com/fffaraz/awesome-cpp/blob/master): 可移植和快速的加密库。
* [Botan](http://botan.randombit.net/): C++加密库。
* [Crypto++](http://www.cryptopp.com/): 一个有关加密方案的免费的C++库。
* [digestpp](https://github.com/kerukuro/digestpp): C++11的信息摘要（哈希）库，只有头文件。
* [GnuPG](https://www.gnupg.org/): OpenPGP标准的完整和免费实现。
* [GnuTLS](http://www.gnutls.org/): 实现了SSL，TLS和DTLS协议的安全通信库。
* [Libgcrypt](http://www.gnu.org/software/libgcrypt/): 通用的密码库，最初基于CnuPG的代码。
* [LibreSSL](http://www.libressl.org/): 免费的SSL/TLS协议，属于2014 OpenSSL的一个分支。
* [libsodium](https://github.com/jedisct1/libsodium): 基于NaCI的加密库，固执己见，容易使用。
* [libhydrogen](https://github.com/jedisct1/libhydrogen): 轻量级、安全、易于使用的加密库，适用于受限的环境。
* [LibTomCrypt](https://github.com/libtom/libtomcrypt): 相当全面、模块化和可移植的加密工具包。
* [mbedTLS](https://github.com/ARMmbed/mbedtls): 开源的、可移植的、易于使用的、可读的、灵活的SSL库，以前被称为PolarSSL。
* [Nettle](http://www.lysator.liu.se/~nisse/nettle/) 底层的加密库。
* [OpenSSL](http://www.openssl.org/): 一个健壮的，商用的，功能齐全的，开源的加密库。
* [retter](https://github.com/MaciejCzyzewski/retter): 与哈希函数、密码、工具、库和材料相关的密码学集合。
* [s2n](https://github.com/awslabs/s2n): TLS/SSL协议的实现。
* [sha1collisiondetection](https://github.com/cr-marcstevens/sha1collisiondetection): 用于检测文件中SHA-1冲突的库和命令行工具。
* [Tink](https://github.com/google/tink): 多语言、跨平台的库，提供安全、易于正确使用且难以滥用的加密API。
* [Tiny AES in C](https://github.com/kokke/tiny-AES-c): C中小型可移植的AES128/192/256。
* [Themis](https://github.com/cossacklabs/themis): 用于无痛数据安全的加密库，为移动和服务器平台提供对称和非对称加密，具有向前保密性的安全套接字。

**CSV**

用于解析逗号分隔值（CSV）文件的库

* [csv2](https://github.com/p-ranav/csv2): 现代C++的快速CSV解析器。
* [Fast C++ CSV Parser](https://github.com/ben-strasser/fast-cpp-csv-parser): 用于读取CSV文件的小型、易于使用和快速的库，只包含头文件。
* [Vince's CSV Parser](https://github.com/vincentlaucsb/csv-parser): 快速、独立、流式的C++17 CSV解析器，具有可选的类型转换和统计信息。

**数据库**

数据库，SQL服务器，ODBC驱动程序和工具

* [hiberlite](https://github.com/paulftw/hiberlite): 用于Sqlite3的C++对象关系映射。
* [Hiredis](https://github.com/redis/hiredis): 用于Redis数据库的很简单的C客户端库。
* [LevelDB](https://github.com/google/leveldb): 由Google编写的快速键值存储库，提供了从字符键到字符串值的有序映射。
* [LMDB](http://symas.com/mdb/): 符合数据库四大基本元素的嵌入键值存储。
* [LMDB++](https://github.com/bendiken/lmdbxx): LMDB嵌入式数据库的C++11封装器。
* [MongoDB C Driver](https://github.com/mongodb/mongo-c-driver): C的MongoDB客户端库。
* [MongoDB C++ Driver](https://github.com/mongodb/mongo-cxx-driver): MongDB的C++驱动程序。
* [MongoDB Libbson](https://github.com/mongodb/libbson): BSON实用库。
* [MySQL++](http://www.tangentsoft.net/mysql++/): 封装了MySql的C API的C++ 封装器。
* [nanodbc](https://github.com/nanodbc/nanodbc): 用于本机C ODBC API的小型C++封装器。
* [ODB](https://www.codesynthesis.com/products/odb/): C++的开源、跨平台、跨数据库的对象关系映射（ORM）系统。
* [redis3m](https://github.com/luca3m/redis3m): 使用干净C++接口的hiredis封装器，支持标记和现成的模式。
* [RocksDB](https://github.com/facebook/rocksdb): 来自Facebook的嵌入键值的快速存储。
* [SimDB](https://github.com/LiveAsynchronousVisualizedArchitecture/simdb): 高性能、共享内存、无锁、跨平台、单文件、最小依赖的C++11键值存储。
* [SOCI](https://github.com/SOCI/soci): C++的数据库抽象层。
* [SQLite](http://www.sqlite.org/): 一个完全嵌入式的，功能齐全的关系数据库，只有几百KB，可以正确包含到你的项目中。
* [SQLiteC++](https://github.com/SRombauts/SQLiteCpp): SQLiteC++ (SQLiteCpp)是聪明且易于使用的C++ SQLite3封装器。
* [sqlite\_modern\_cpp](https://github.com/SqliteModernCpp/sqlite_modern_cpp): sqlite库的C++14封装器，只有头文件。
* [sqlite\_orm](https://github.com/fnc12/sqlite_orm): 适用于现代C++的SQLite ORM轻量库，只有头文件。
* [sqlpp11](https://github.com/rbock/sqlpp11): 用于SQL查询和在C++产生结果的类型安全的嵌入式域特定语言。
* [TileDB](https://github.com/TileDB-Inc/TileDB): 快速密集和稀疏多维数组DBMS。
* [UnQLite](https://github.com/symisc/unqlite): 独立、无服务器、零配置的事物型NoSQL引擎。
* [upscaledb](https://upscaledb.com/): 具有内置查询接口的嵌入式“typed”键值存储。

**调试**

调试库， 内存和资源泄露检测，单元测试

* [backward-cpp](https://github.com/bombela/backward-cpp): C++的漂亮堆栈轨迹打印机。
* [benchmark](https://github.com/google/benchmark): Google提供的小型微基准支持库。
* [Boost.Test](http://www.boost.org/doc/libs/master/libs/test/doc/html/index.html): Boost测试库。
* [check](https://github.com/libcheck/check): C的单元测试框架。
* [doctest](https://github.com/onqtam/doctest): 功能最轻便的单个C++头文件的测试框架。
* [Catch](https://github.com/philsquared/Catch): 一个很时尚的，C++原生的框架，只包含头文件，用于单元测试，测试驱动开发和行为驱动开发。
* [Catch2](https://github.com/catchorg/Catch2): 一个很时尚的，C++原生的框架，只包含头文件，用于单元测试，测试驱动开发和行为驱动开发。
* [Celero](https://github.com/DigitalInBlue/Celero): C++基准测试框架。
* [CppUTest](https://github.com/cpputest/cpputest): C/C++的单元测试和模拟框架。
* [CUTE](http://cute-test.com/): 更容易的C++单元测试。
* [CMocka](https://cmocka.org/): 支持模拟对象的C单元测试框架。
* [CppUnit](http://www.freedesktop.org/wiki/Software/cppunit/): 由JUnit移植过来的C++测试框架。
* [CTest](http://www.cmake.org/cmake/help/v2.8.8/ctest.html): CMake测试驱动程序。
* [dbg-macro](https://github.com/sharkdp/dbg-macro): C++的dbg（……）宏。
* [Deleaker](http://www.deleaker.com/): 用于资源泄露检查的工具，包括内存、GDI和处理泄露。
* [fff](https://github.com/meekrosoft/fff): 创建伪C函数的微框架。
* [Google Mock](https://github.com/google/googletest/blob/master/googlemock/README.md): 编写和使用C++模拟类的库。
* [Google Test](https://github.com/google/googletest): Google的C++测试框架。
* [ig-debugheap](https://github.com/deplinenoise/ig-debugheap): 用于追踪内存错误的多平台调试堆。
* [libtap](https://github.com/zorgnax/libtap): 用C语言编写测试。
* [MemTrack](http://www.almostinfinite.com/memtrack.html): 用于C++跟踪内存分配。
* [microprofile](https://bitbucket.org/jonasmeyer/microprofile/overview): 跨平台的网络视图分析器。
* [minUnit](http://www.jera.com/techinfo/jtns/jtn002.html): 使用C写的最小的单元测试框架，包含在单个头文件中。
* [Mockator](http://www.mockator.com/): 用于C++缝合和模拟对象的Eclipse CDT插件。
* [Nanotimer](https://github.com/mattreecebentley/plf_nanotimer): 用于基准测试的简单、低开销、跨平台的定时器类。
* [Nonius](https://github.com/libnonius/nonius): C++微基准测试框架。
* [Remotery](https://github.com/Celtoys/Remotery): 用于Web视图的单一C文件分析器。
* [UnitTest++](http://unittest-cpp.sourceforge.net/): 轻量级的C++单元测试框架。
* [Unity](https://github.com/ThrowTheSwitch/Unity): C的简单单元测试。
* [utest.h](https://github.com/sheredom/utest.h): C和C++的单个头文件测试框架。
* [μt](https://github.com/boost-experimental/ut): C++20的单头文件/单模块、无宏的μ（微型）/单元测试框架。
* [VLD](http://vld.codeplex.com/): 微软测漏仪，Visual C++的免费、健壮、开源的内存泄漏检测系统。

**字体**

解析和处理字体文件的库。

* [Fontconfig](https://gitlab.freedesktop.org/fontconfig/fontconfig): 字体配置和自定义库。
* [FreeType](https://www.freetype.org/): 用于渲染字体的免费、可获得的软件库。
* [otfcc](https://github.com/caryll/otfcc): 用于解析和编写OpenType字体文件的C库和实用程序。

**游戏引擎**

* [Acid](https://github.com/Equilibrium-Games/Acid): 高速的C++17 Vulkan游戏引擎。
* [Allegro](http://liballeg.org/): 主要用于视频游戏和多媒体编程的跨平台库。
* [Cocos2d-x](http://www.cocos2d-x.org/): 用于构建2D游戏、交互式书籍、演示和其它图形应用程序的多平台框架。
* [Corange](https://github.com/orangeduck/Corange): 用纯C、SDL、OpenGL编写的游戏引擎。
* [Crown](https://github.com/dbartolini/crown): Crown是一款通用的数据驱动游戏引擎，使用正统C++从零开始编写，并牢记简约和面向数据的设计理念。
* [delta3d](http://sourceforge.net/projects/delta3d/): 健壮的仿真平台。
* [EnTT](https://github.com/skypjack/entt): 游戏与现代C++的碰撞。
* [GamePlay](https://github.com/gameplay3d/GamePlay): 用于创建2D/3D移动和桌面游戏的跨平台本机C++游戏框架。
* [Godot](https://github.com/godotengine/godot): 功能齐全、开源、有MIT许可的游戏引擎。
* [Grit](http://gritengine.com/): 社区项目，用于构建一个免费的游戏引擎，实现开放的世界3D游戏。
* [Halley](https://github.com/amzeratul/halley): 用C++14编写的轻量级游戏引擎，带有“真实的”实体组件系统。
* [KlayGE](https://github.com/gongminmin/KlayGE): 具有基于插件架构的跨平台、开源的游戏引擎。
* [nCine](https://github.com/nCine/nCine): 注重性能的跨平台2D游戏引擎，用C++11编写，还可以选择使用Lua编写脚本。
* [OpenXRay](https://github.com/OpenXRay/xray-16): 在S.T.A.L.K.E.R.游戏系列中使用的社群修改的X射线引擎。
* [Oxygine](http://oxygine.org/): 跨平台的2D C++游戏引擎。
* [Panda3D](https://github.com/panda3d/panda3d): 游戏引擎，用于Python和C++程序的3D渲染和游戏开发的框架。
* [PixelGameEngine](https://github.com/OneLoneCoder/olcPixelGameEngine): olcPixelGameEngine的官方发行版，该工具用于javidx9的YouTube视频和项目。
* [Polycode](http://polycode.org/): C++实现的用于创建游戏的开源框架（与Lua绑定）。
* [raylib](https://github.com/raysan5/raylib): 简单、易于使用的库，用于享受视频游戏编程的乐趣。
* [Spring](https://github.com/spring/spring): 强大、免费、跨平台的RTS游戏引擎。
* [Torque2D](https://github.com/GarageGames/Torque2D): 为2D游戏开发构建的开源、跨平台C++引擎。
* [Torque3D](https://github.com/GarageGames/Torque3D): 为3D游戏开发构建的开源C++引擎。
* [toy engine](https://github.com/hugoam/toy): toy是一个轻量的模块化的 C++ 游戏引擎，并提供简单的、有表现力的C++惯用法来快速迭代设计功能齐全的2D和3D游戏。
* [Urho3D](https://urho3d.github.io/): C++实现的免费、轻量级、跨平台的2D和3D游戏引擎，受到OGRE和Horde3D很大的启发。

**图形用户界面**

图形用户界面

* [Boden](https://github.com/AshampooSystems/boden): 本机、移动、跨平台的GUI框架。
* [CEGUI](http://cegui.org.uk/): 灵活的跨平台GUI库。
* [Elements](https://github.com/cycfi/elements): 轻量级、细粒度、分辨率无关的模块化GUI库。
* [FLTK](http://www.fltk.org/index.php): 快速，轻量级的跨平台的C++GUI工具包。
* [GacUI](https://github.com/vczh-libraries/GacUI): GPU加速的C++用户界面，具有所见即所得的开发工具，支持XML，具有内置数据绑定和MVVM功能。
* [GTK+](http://www.gtk.org/): 用于创建图形用户界面的多平台工具包。
* [gtkmm](http://www.gtkmm.org/en/): 用于受欢迎的GUI库GTK+的官方C++接口。
* [imgui](https://github.com/ocornut/imgui): 拥有最小依赖关系的即时模式图形用户界面。
* [implot](https://github.com/epezent/implot): imgui的即时模式绘图控件。
* [iup](https://www.tecgraf.puc-rio.br/iup): 构建图形用户界面的多平台工具包。
* [libui](https://github.com/andlabs/libui): C中简单、可移植（但是不灵活）的GUI库，使用其支持的每一种平台的本机GUI技术。
* [MyGUI](http://mygui.info/): 快速，灵活，简单的GUI。
* [nana](http://nanapro.org/en-us/): Nana是现代C++风格的GUI编程的跨平台库。
* [NanoGui](https://github.com/mitsuba-renderer/nanogui): 简约、跨平台的控件库，用于OpenGL 3.x及更高版本。
* [nuklear](https://github.com/Immediate-Mode-UI/Nuklear): 单个头文件的ANSI C gui库。
* [QCustomPlot](http://qcustomplot.com/): 没有更多依赖关系的Qt绘图控件。
* [Qwt](http://qwt.sourceforge.net/): 用户与技术应用的Qt 控件。
* [QwtPlot3D](http://qwtplot3d.sourceforge.net/): 功能丰富的基于Qt/OpenGL的C++编程库，本质上提供了一群3D控件。
* [RmlUi](https://github.com/mikke89/RmlUi): 进化的HTML/CSS用户界面库，libRocket的分支。
* [Sciter](http://sciter.com/): Sciter是可嵌入的HTML/CSS/脚本语言引擎，旨在用于现代桌面应用程序的UI层。
* [wxWidgets](http://wxwidgets.org/) C++库，允许开发人员使用一个代码库可以为widows， Mac OS X，Linux和其他平台创建应用程序。
* [Yue](https://github.com/yue/yue): 创建本机跨平台GUI应用的库。

**图形**

* [assimp](https://github.com/assimp/assimp): 开放资产导入库（assimp）是跨平台的3D模型导入库，旨在为不同的3D资产文件格式提供通用的API。
* [bgfx](https://github.com/bkaradzic/bgfx): 跨平台的渲染库。
* [Blend2D](https://github.com/blend2d/blend2d): 由JIT编译器提供技术支持的2D矢量图形引擎。
* [bs::framework](https://github.com/GameFoundry/bsf): 用于实时图形应用开发的现代C++14库。
* [Cairo](http://www.cairographics.org/): 支持多种输出设备的2D图形库。
* [C-Turtle](https://github.com/walkerje/C-Turtle): 充当Clmg封装器的C++11 turtle图形库，只有头文件。
* [Diligent Engine](https://github.com/DiligentGraphics/DiligentEngine): 现代、跨平台、低阶的3D图形库。
* [DirectXTK](https://github.com/Microsoft/DirectXTK): 用于在C++中编写DirectX 11.x代码的帮助类的集合。
* [GLFW](https://github.com/glfw/glfw): 简单、跨平台的OpenGL处理库。
* [herebedragons](https://github.com/kosua20/herebedragons): 使用各种引擎、框架或者API实现的基本3D场景。
* [Horde3D](https://github.com/horde3d/Horde3D) 一个小型的3D渲染和动画引擎。
* [Ion](https://github.com/google/ion): 小型高效的库集合，用于构建使用3D图形的跨平台客户端或服务器应用程序。
* [Irrlicht](http://irrlicht.sourceforge.net/): C++编写的高性能实时3D引擎。
* [libigl](https://github.com/libigl/libigl): 简单的C++几何处理库。
* [LLGL](https://github.com/LukasBanana/LLGL): 低水平图形库（LLGL）是现代图形API的薄抽象层。
* [magnum](https://github.com/mosra/magnum): 用于游戏和数据可视化的轻量级和模块化的 C++ 11 /C++ 14图形中间件。
* [NanoVG](https://github.com/memononen/nanovg): OpenGL之上的抗锯齿2D矢量图形库，用于UI和可视化。
* [Ogre 3D](http://www.ogre3d.org/): 用C++编写的一个面向场景，实时，灵活的3D渲染引擎（并非游戏引擎）。
* [OpenSceneGraph](http://www.openscenegraph.org/): 具有高性能的开源3D图形工具包。
* [OpenSubdiv](https://github.com/PixarAnimationStudios/OpenSubdiv): Pixar的库，用于评估和渲染CPU和GPU上的细分曲面。
* [OpenVDB](http://www.openvdb.org/): 用于存储、编辑、渲染体积数据库的库和工具。
* [Panda3D](http://www.panda3d.org/): 用于3D渲染和游戏开发的框架，用Python和C++编写。
* [Partio](https://github.com/wdas/partio): 用于处理粒子数据的库，支持大多数常见文件形式。
* [Skia](https://github.com/google/skia) 用于绘制文字，图形和图像的完整的2D图形库。
* [TinySpline](https://github.com/msteinbeck/tinyspline): 小型但强大的ANSI C库，用于差值、转换和查询任意NURBS、B-样条函数和贝赛尔曲线。
* [urho3d](https://github.com/urho3d/Urho3D) 跨平台的渲染和游戏引擎。
* [Yocto/GL](https://github.com/xelatihy/yocto-gl): 用于数据驱动的基于物理图形的微型C++库。

**图像处理**

* [Boost.GIL](http://www.boost.org/doc/libs/1_56_0/libs/gil/doc/index.html): 通用图像库。
* [CImg](http://cimg.sourceforge.net/): 用于图像处理的小型开源C++工具包。
* [CxImage](http://www.xdp.it/cximage.htm): 用于加载，保存，显示和转换的图像处理和转换库，可以处理的图片格式包括 BMP, JPEG, GIF, PNG, TIFF, MNG, ICO, PCX, TGA, WMF, WBMP, JBG, J2K。
* [Dlib](https://github.com/davisking/dlib): 现代C++11的机器学习、计算机视觉、数值优化和深度学习工具包。
* [FreeImage](http://freeimage.sourceforge.net/): 开源库，支持现在多媒体应用所需的通用图片格式和其他格式。
* [GD](https://github.com/libgd/libgd): GD图形库，有名的用于PHP中图形加载/操作和缩略图生成。
* [DCMTK](http://dicom.offis.de/dcmtk.php.en): DICOM工具包。
* [GDCM](http://gdcm.sourceforge.net/wiki/index.php/Main_Page): Grassroots DICOM 库。
* [ITK](http://www.itk.org/): 跨平台的开源图像分析系统。
* [Leptonica](https://github.com/DanBloomberg/leptonica): Leptonica是一个包含软件的开源库，其广泛应用于图像处理和图形分析应用程序。
* [libfacedetection](https://github.com/ShiqiYu/libfacedetection): 用于图像中人脸检测的开源库，其人脸检测的速度能达到1500FPS。
* [libjpeg-turbo](https://github.com/libjpeg-turbo/libjpeg-turbo): 使用SIMD指令来加速基线JPEG编码和解码的JPEG图像编解码器。
* [libvips](https://github.com/jcupitt/libvips): 具有低内存需求的快速图像处理库。
* [Magick++](http://www.imagemagick.org/script/api.php): ImageMagick程序的C++接口。
* [MagickWnd](http://www.imagemagick.org/script/api.php): ImageMagick程序的C++接口。
* [OpenCV](http://opencv.org/): 开源计算机视觉类库。
* [OpenEXR](http://www.openexr.com/): 具有高动态范围成像的跨平台库。
* [OpenImageIO](https://github.com/OpenImageIO/oiio): 强大的图像和纹理处理库，支持多种常见的有损和RAW格式。
* [Simd](https://github.com/ermig1979/Simd): 使用SIMD的C++图像处理库: SSE，SSE2，SSE3，SSSE3，SSE4.1，SSE4.2，AVX，AVX2，AVX-512，VMX（Altivec）和VSX（Power7），NEON for ARM。
* [stb-image](https://github.com/nothings/stb/blob/master/stb_image.h): STB单个头文件的图像加载库。
* [tesseract-ocr](https://code.google.com/p/tesseract-ocr/): OCR引擎。
* [TinyEXIF](https://github.com/cdcseacave/TinyEXIF): 用于JPEG的微型ISO兼容的C++ EXIF和XMP解析库。
* [Video++](https://github.com/matt-42/vpp): C++ 14高性能的视频和图像处理库。
* [VIGRA](https://github.com/ukoethe/vigra): 用于图像分析通用C++计算机视觉库。
* [VTK](http://www.vtk.org/): 用于3D计算机图形学，图像处理和可视化的开源免费软件系统。

**国际化**

* [gettext](http://www.gnu.org/software/gettext/): GNU `gettext'。
* [IBM ICU](http://site.icu-project.org/): 提供Unicode 和全球化支持的C、C++ 和Java库。
* [libiconv](http://www.gnu.org/software/libiconv/): 用于不同字符编码之间的编码转换库。
* [utf8.h](https://github.com/sheredom/utf8.h): C和C++的单个头文件的utf8字符串函数。
* [utf8proc](https://github.com/JuliaStrings/utf8proc): 干净的C库，用于处理UTF-8 Unicode数据。

**进程间通信**

* [Apache Thrift](https://thrift.apache.org/): 高效的跨语言IPC/RPC，可在C++、Java、Python、PHP、C#和许多其它语言之间运行。最初是由Facebook开发。
* [Cap'n Proto](https://github.com/capnproto/capnproto): 快速的数据交换格式和基于功能的RPC系统。
* [eCAL](https://github.com/continental/ecal): 发布/订阅，客户端/服务器，C++/Python/C#，各种消息协议（protobuf, capnproto ..）。
* [gRPC](https://github.com/grpc/grpc): 高性能、开源、通用的RPC框架。
* [Ice](https://github.com/zeroc-ice/ice): 全面的RPC框架，支持C++、C＃、Java、JavaScript、Python等。
* [libjson-rpc-cpp](https://github.com/cinemast/libjson-rpc-cpp): 用于C++服务器和客户端的JSON-RPC框架。
* [nanomsg](https://github.com/nanomsg/nanomsg): 好几个“可扩展协议”的简单高性能实现。
* [nng](https://github.com/nanomsg/nng): nanomsg的下一代，轻量级、无代理消息的传递库。
* [rpclib](https://github.com/rpclib/rpclib): 现代C++的msgpack-RPC服务器和客户端库。
* [simple-rpc-cpp](https://github.com/pearu/simple-rpc-cpp): C / C++函数的简单RPC封装生成器。
* [WAMP](http://wamp.ws/): 提供RPC和发布/订阅消息传递模式。（各种实现，各种语言）
* [xmlrpc-c](http://xmlrpc-c.sourceforge.net/): 基于XML和HTTP的轻量级RPC库。

**Json**

* [Boost.PropertyTree](http://www.boost.org/doc/libs/1_55_0/doc/html/property_tree.html): 属性树解析器/生成器，可用于解析XML / JSON / INI / Info文件。
* [cJSON](https://github.com/DaveGamble/cJSON): ANSI C的超轻量级JSON解析器。
* [frozen](https://github.com/cesanta/frozen): C/C++的Json解析生成器。
* [Jansson](https://github.com/akheron/jansson): 进行编解码和处理Json数据的C语言库。
* [jbson](https://github.com/chrismanning/jbson): C++14中构建和迭代BSON data和Json 文档的库。
* [JeayeSON](https://github.com/jeaye/jeayeson): 非常健全的C++ JSON库，只包含头文件。
* [Jsmn](https://github.com/zserge/jsmn): C语言中的简约JSON解析器。
* [json](https://github.com/nlohmann/json): 用于现代C++的JSON。
* [JSON++](https://github.com/hjiang/jsonxx): C++的JSON 解析器。
* [json.h](https://github.com/sheredom/json.h): 在C和C++中解析JSON的简单的单头文件/单源文件的解决方案。
* [json-c](https://github.com/json-c/json-c): C中的JSON实现。
* [jsoncons](https://github.com/danielaparker/jsoncons): 使用JSONPointer、JSONPatch、JSONPath和JMESPath的JSON和类似JSON二进制格式的C++库，只有头文件。
* [JsonCpp](https://github.com/open-source-parsers/jsoncpp): 可与JSON交互的C++库。
* [json-parser](https://github.com/udp/json-parser): 用可移植的ANSI C编写的JSON解析器，占用内存非常少。
* [json11](https://github.com/dropbox/json11): 一个迷你的C++11 JSON库。
* [json-voorhees](https://github.com/tgockel/json-voorhees): C++的JSON库。支持C++ 11，无依赖关系，快速且对开发人员友好。
* [jute](https://github.com/amir-s/jute): 非常简单的C++ JSON解析器。
* [ibjson](https://github.com/vincenthz/libjson): C语言中的JSON解析和打印库，很容易和任何模型集成。。
* [libjson](http://sourceforge.net/projects/libjson/): 轻量级的JSON库。
* [LIBUCL](https://github.com/vstakhov/libucl): 通用配置库解析器。
* [parson](https://github.com/kgabis/parson): 用C编写的轻量级的JSON库。
* [PicoJSON](https://github.com/kazuho/picojson): C++中JSON解析序列化，只包含头文件。
* [qt-json](https://github.com/gaudecker/qt-json): 用于JSON数据和 QVariant层次间的相互解析的简单类。
* [QJson](https://github.com/flavio/qjson): 将JSON数据映射到QVariant对象的基于Qt的库。
* [RapidJSON](https://github.com/miloyip/rapidjson): 用于C++的快速JSON 解析生成器，包含SAX和DOM两种风格的API。
* [sajson](https://github.com/chadaustin/sajson): 用于C++ 11的轻量级、高性能的JSON解析器。
* [simdjson](https://github.com/lemire/simdjson): 极快的JSON库，可以千兆字节每秒的速度解析JSON。
* [taoJSON](https://github.com/taocpp/json): 零依赖的C++JSON库，只包含头文件。
* [ujson](https://bitbucket.org/awangk/ujson): µjson是一个小型、C++ 11、UTF-8、JSON的库。
* [YAJL](https://github.com/lloyd/yajl): C语言中快速流JSON解析库。

**日志**

* [Blackhole](https://github.com/3Hren/blackhole): 基于属性的日志框架，旨在快速、模块化和高度可定制。
* [Boost.Log](http://www.boost.org/doc/libs/1_56_0/libs/log/doc/html/index.html): 设计非常模块化，并且具有扩展性。
* [Easylogging++](https://github.com/easylogging/easyloggingpp): 适用于C++ 11（或更高版本）应用程序的超轻量级、高性能的日志库。
* [G3log](https://github.com/KjellKod/g3log): 具有动态接收模块的异步记录器。
* [glog](https://github.com/google/glog): Google日志模块的C++实现。
* [Log4cpp](http://log4cpp.sourceforge.net/): C++类库，用于灵活地添加日志到文件，系统日志，IDSA和其他地方。
* [log4cplus](https://github.com/log4cplus/log4cplus): 易于使用的C++日志API，可提供对日志管理和配置的线程安全、灵活且任意粒度的控制。
* [loguru](https://github.com/emilk/loguru): 轻量级的C++日志库。
* [plog](https://github.com/SergiusTheBest/plog): C++可移植且简单的日志，少于1000行代码。
* [reckless](https://github.com/mattiasflodin/reckless): C++低延迟、高吞吐量、异步的日志库。
* [spdlog](https://github.com/gabime/spdlog): 超级快的C++日志库，只包含头文件。
* [templog](http://www.templog.org/): 小型、轻量级的 C++ 库，可以添加日志到你的C++应用程序中。
* [P7Baical](http://baical.net/p7.html): 开源、跨平台的库，用于在最小限度使用CPU和内存的情况下高速发送遥测和跟踪数据。
* [Quill](https://github.com/odygrd/quill): 异步、跨平台、低时延的日志库。

**机器学习**

* [Caffe](https://github.com/BVLC/caffe): 快速的神经网络框架。
* [CCV](https://github.com/liuliu/ccv): 基于C /缓存/核心的计算机视觉库，现代计算机视觉库。
* [darknet](https://github.com/pjreddie/darknet): 用C和CUDA编写的开源神经网络框架。
* [Dlib](https://github.com/davisking/dlib): 现代C++ 11的机器学习、计算机视觉、数值优化和深度学习的工具包。
* [Fido](https://github.com/FidoProject/Fido): 用于嵌入式电子和机器人技术的高度模块化的C++机器学习库。
* [flashlight](https://github.com/facebookresearch/flashlight): 来自Facebook AI研究中心的快速、灵活的机器学习库，其完全用C++编写，基于ArrayFire张量库。
* [libsvm](https://github.com/cjlin1/libsvm): 支持向量机的简单、易用、高效的库。
* [MeTA](https://github.com/meta-toolkit/meta): 现代C++的数据科学工具包。
* [Minerva](https://github.com/dmlc/minerva): 快速、灵活的深度学习系统。
* [mlpack](http://www.mlpack.org/): 可扩展的C++机器学习库。
* [OpenCV](https://github.com/Itseez/opencv): 开源计算机视觉库。
* [Recommender](https://github.com/GHamrouni/Recommender): 使用协同过滤进行产品推荐/建议的C语言库。
* [RNNLIB](https://github.com/szcom/rnnlib): RNNLIB是用于序列学习问题的递归神经网络库。
* [SHOGUN](https://github.com/shogun-toolbox/shogun): Shogun 机器学习工具。
* [sofia-ml](https://code.google.com/p/sofia-ml/): 用于机器学习的快速增量算法套件。
* [VLFeat](https://github.com/vlfeat/vlfeat): VLFeat开源库实现了流行的计算机视觉算法，这些算法专门用于图像理解以及局部特征提取和匹配。
* [xgboost](https://github.com/dmlc/xgboost): 用于Python、R、Java、Scala、C++等的可扩展、便携式和分布式梯度增强（GBDT，GBRT或GBM）的库。在单机、Hadoop、Spark、Flink和DataFlow上运行。

**数学**

* [Apophenia](https://github.com/b-k/apophenia): 用于统计和科学计算的C库。
* [Armadillo](http://arma.sourceforge.net/): 用于线性代数和科学计算的快速C++库。
* [autodiff](https://github.com/autodiff/autodiff): 用于自动微分的现代、快速且富有表现力的C++库。
* [blaze](https://code.google.com/p/blaze-lib/): 高性能的C++数学库，用于密集和稀疏算法。
* [Boost.Multiprecision](http://www.boost.org/doc/libs/master/libs/multiprecision/doc/html/index.html): 在C++中提供更高范围/精度的整数、有理数和浮点数类型，只包含头文件或者在GMP / MPFR / LibTomMath后端。
* [ceres-solver](http://ceres-solver.org/): 来自谷歌的C++库，用于建模和解决大型复杂非线性最小平方问题。
* [CGAL](https://github.com/CGAL/cgal): 高效，可靠的几何算法集合。
* [cml](http://cmldev.net/): 用于游戏和图形的免费C++数学库。
* [Dlib](https://github.com/davisking/dlib): 现代C++11的机器学习、计算机视觉、数值优化和深度学习工具包。
* [Eigen](http://eigen.tuxfamily.org/): 高级C++模板头文件库，包括线性代数，矩阵，向量操作，数值解决和其他相关的算法。
* [ExprTK](http://www.partow.net/programming/exprtk/): C++数学表达式工具包库（ExprTk）是一个易于使用、易于集成且非常高效的运行时数学表达式解析器和评估引擎。
* [Geometric Tools](https://www.geometrictools.com/): 用于数学、图形、图像分析和物理学领域计算的C++库。
* [GLM](https://github.com/g-truc/glm): 只包含头文件的C++数学库，可与OpenGL的GLSL数学匹配并互操作。
* [GMTL](http://ggt.sourceforge.net/): 数学图形模板库是一组广泛实现基本图形的工具。
* [GMP](https://gmplib.org/): 用于个高精度计算的C/C++库，处理有符号整数，有理数和浮点数。
* [Klein](https://github.com/jeremyong/klein): 快速、SIMD优化的C++ 17几何代数库，用于点、线和平面投影、相交、连接、刚体运动等。
* [linalg.h](https://github.com/sgorsten/linalg): C++的单头文件、公共域、短向量数学库。
* [MIRACL](https://github.com/CertiVox/MIRACL): 多精度整数和有理数算法加密库。
* [muparser](http://beltoforion.de/article.php?a=muparser): muParser是用C++编写的可扩展、高性能的数学表达式解析库。
* [LibTomMath](https://github.com/libtom/libtommath): 完全用C编写的免费、开源、可移植的数论多精度整数库。
* [linmath.h](https://github.com/datenwolf/linmath.h): 精益线性数学库，主要用于图形编程。
* [lp\_solve](https://sourceforge.net/projects/lpsolve): 用于制定和解决线性编程问题的库。
* [OpenBLAS](https://github.com/xianyi/OpenBLAS): 基于GotoBLAS2 1.13 BSD版本的优化BLAS库。
* [QuantLib](https://github.com/lballabio/quantlib)v: 用于计量金融的免费/开源库。
* [StatsLib](https://github.com/kthohr/stats): 统计分布函数的C++库，只包含头文件。
* [SymEngine](https://github.com/symengine/symengine): 快速符号处理库，用C++重写了SymPy的核心。
* [TinyExpr](https://github.com/codeplea/tinyexpr): 用于解析和评估字符串中的数学表达式的C库。
* [Vc](https://github.com/VcDevel/Vc): C++的SIMD矢量类。
* [Versor](http://versor.mat.ucsb.edu/): （快速的）几何代数的通用C++库，包括欧几里得、投影，保角、时空等。
* [Wykobi](http://www.wykobi.com/): 高效、健壮且易于使用的C++库，面向C++ 2D / 3D的计算几何例程。
* [xtensor](https://github.com/QuantStack/xtensor): 受NumPy语法启发的C++ 14库，用于使用多维数组表达式进行数值分析。
* [universal](https://github.com/stillwater-sc/universal): 只包含头文件的C++ 14库，实现任意假定算数。假定数值系统是一个锥形浮点，比IEEE浮点更有效。假定数使可复现的计算科学成为可能。

**内存分配**

* [Boehm GC](https://github.com/ivmai/bdwgc): C和C++的保守的垃圾回收器。
* [C Smart Pointers](https://github.com/Snaipe/libcsptr): （GNU）C编程语言的智能指针。
* [Hoard](https://github.com/emeryberger/Hoard): 快速、可扩展和节约内存的Malloc，支持Linux、Windows和Mac。
* [jemalloc](https://github.com/jemalloc/jemalloc): 通用的malloc（3）实现，强调避免存储碎片和可扩展的并发支持。
* [memory](https://github.com/foonathan/memory): 兼容STL的C++内存分配器库。
* [memory-allocators](https://github.com/mtrebi/memory-allocators): 自定义内存分配器，可提高动态内存分配的性能。
* [mimalloc](https://github.com/microsoft/mimalloc): 具有出色性能的紧凑型通用分配器。
* [tgc](https://github.com/orangeduck/tgc): 用 ~500 LOC编写的C语言微型垃圾回收器。

**多媒体**

* [GStreamer](http://gstreamer.freedesktop.org/): 构建媒体处理组件图形的库。
* [libass](https://github.com/libass/libass): ASS / SSA字幕格式的便携式字幕渲染器。
* [libass](https://github.com/libass/libass): 处理多媒体内容（例如音频、视频、字幕和相关元数据）的库和工具的集合。
* [LIVE555 Streaming Media](http://www.live555.com/liveMedia/): 使用开放标准协议(RTP/RTCP, RTSP, SIP) 的多媒体流库。
* [libVLC](https://wiki.videolan.org/LibVLC): libVLC (VLC SDK)媒体框架。
* [MediaInfoLib](https://github.com/MediaArea/MediaInfoLib): 用于视频和音频文件中最相关的技术和标签数据的方便统一显示。
* [QtAv](https://github.com/wang-bin/QtAV): 基于Qt和FFmpeg的多媒体播放框架，能够帮助你轻而易举地编写出一个播放器。
* [SDL](http://www.libsdl.org/): 简单直控媒体层。
* [SFML](http://www.sfml-dev.org/): 快速，简单的多媒体库。
* [TagLib](https://github.com/taglib/taglib): 用于读取和编辑几种流行音频格式的元数据的库。

**网络**

* [ACE](http://www.cs.wustl.edu/~schmidt/ACE.html): C++面向对象网络编程工具包。
* [Boost.Asio](http://think-async.com/): 用于网络和底层I/O编程的跨平台的C++库。
* [Boost.Beast](https://github.com/boostorg/beast): 在C++ 11中基于Boost.Asio构建的HTTP和WebSocket。
* [Breep](https://github.com/Organic-Code/Breep): 基于事件的高级C++ 14点对点库。
* [C++ REST SDK](https://github.com/Microsoft/cpprestsdk): C++ REST SDK（以前称为Casablanca）。
* [Restbed](https://github.com/corvusoft/restbed): C++ 11异步RESTful框架。
* [Restinio](https://github.com/Stiffstream/restinio): 只有头文件的C++ 14库，为您提供嵌入式HTTP / Websocket服务器。
* [c-ares](https://github.com/c-ares/c-ares): 用于异步DNS请求的C库。
* [cpp-httplib](https://github.com/yhirose/cpp-httplib): 单文件的C++ 11 HTTP / HTTPS服务器库，只包含头文件。
* [cpp-netlib](http://cpp-netlib.org/): 高级网络编程的开源库集合。
* [cpp-netlib/uri](https://github.com/cpp-netlib/uri): C++的URI解析器/生成器库，与RFC 3986和RFC 3987兼容。
* [cpr](https://github.com/whoshuu/cpr): 现代C++ HTTP请求库，具有简单但功能强大的接口。模仿Python Requests模块。
* [curlcpp](https://github.com/JosephP91/curlcpp): CURL（libcurl）的面向对象C++封装器。
* [DPDK](https://github.com/DPDK/dpdk): 用于快速打包处理的数据平面开发工具、库和驱动程序。
* [Dyad.c](https://github.com/rxi/dyad): C的异步网络。
* [ENet](https://github.com/lsalzman/enet): 可靠的UDP网络库。
* [evpp](https://github.com/Qihoo360/evpp): 具有TCP / UDP / HTTP协议的C++高性能网络。
* [H2O](https://github.com/h2o/h2o): 优化的HTTP服务器，支持HTTP / 1.x和HTTP / 2。它也可以用作库。
* [HTTP Parser](https://github.com/nodejs/http-parser): C的http请求/响应解析器。
* [KCP](https://github.com/skywind3000/kcp/blob/master/README.en.md): 快速、可靠的ARQ协议，可帮助应用程序减少网络延迟。
* [libcurl](http://curl.haxx.se/libcurl/): 多协议文件传输库。
* [libhttpserver](https://github.com/etr/libhttpserver): 用于创建嵌入式Rest HTTP服务器（以及更多）的C++库。
* [Libmicrohttpd](http://www.gnu.org/software/libmicrohttpd/): GNU libmicrohttpd是一个小型C库，可以轻松地将HTTP服务器作为另一个应用程序的一部分运行。
* [libpcap](https://github.com/the-tcpdump-group/libpcap): 用于网络流量捕获的可移植C / C++库。
* [libquic](https://github.com/devsisters/libquic): 从Chromium的QUIC实现中提取的QUIC协议库。
* [librdkafka](https://github.com/edenhill/librdkafka): 用于C和C++的Apache Kafka客户端库。
* [libwebsockets](https://github.com/warmcat/libwebsockets): 提供客户端和服务器库的轻量级纯C WebSocket实现。
* [lwIP](http://savannah.nongnu.org/projects/lwip/): 轻量级TCP / IP堆栈。
* [Mongoose](https://github.com/cesanta/mongoose): 非常轻量级的网络服务器。
* [MQTT-C](https://github.com/LiamBindle/MQTT-C): 适用于嵌入式系统和类PC机的便携式MQTT C客户端。
* [mTCP](https://github.com/mtcp-stack/mtcp): 用于多核系统的高度可扩展的用户级TCP堆栈。
* [Muduo](https://github.com/chenshuo/muduo): 用于Linux多线程服务器的C++非阻塞网络库。
* [nghttp2](https://github.com/nghttp2/nghttp2): HTTP / 2 C库。
* [Onion](https://github.com/davidmoreno/onion): C语言HTTP服务器库，其设计为轻量级，易使用。
* [PF\_RING™](https://github.com/ntop/PF_RING): 高速打包处理框架。
* [PicoHTTPParser](https://github.com/h2o/picohttpparser): 微小、原始、快速的HTTP请求/响应解析器。
* [POCO](https://github.com/pocoproject): 用于构建网络和基于互联网应用程序的C++类库，可以运行在桌面，服务器，移动和嵌入式系统。
* [Proxygen](https://github.com/facebook/proxygen): Facebook的C++ HTTP库集合，包括易于使用的HTTP服务器。
* [RakNet](https://github.com/OculusVR/RakNet): 为游戏开发人员提供的跨平台的开源C++网络引擎。
* [restclient-cpp](https://github.com/mrtazz/restclient-cpp): 用于C++的简单REST客户端。它包装libcurl用于HTTP请求。
* [Seasocks](https://github.com/mattgodbolt/seasocks): 具有WebSockets支持的简单、小型的C++嵌入式Web服务器。
* [Silicon](http://siliconframework.org/): 高性能、面向中间件的C++ 14 http Web框架。
* [tlse](https://github.com/eduardsui/tlse): 单个C文件的TLS 1.2 / 1.3实现，使用tomcrypt作为加密库。
* [Tufão](https://github.com/vinipsmaker/tufao): 基于Qt构建的C++异步Web框架。
* [uriparser](https://github.com/uriparser/uriparser): 严格符合RFC 3986的URI解析和处理库。
* [uWebSockets](https://github.com/uNetworking/uWebSockets): µWS是最轻量级、高效和可伸缩的WebSocket和HTTP服务器的实现之一。
* [WAFer](https://github.com/riolet/WAFer): 基于C语言的超轻型软件平台，用于可扩展的服务器端和网络应用。为C程序员考虑了node.js。
* [Wangle](https://github.com/facebook/wangle): 客户端/服务器应用程序框架，用于构建异步的、事件驱动的现代C++服务。
* [wdt](https://github.com/facebook/wdt): 可嵌入式库（和命令行工具），旨在通过多个TCP路径尽可能快地在2个系统之间传输数据。
* [WebSocket++](https://github.com/zaphoyd/websocketpp): 基于C++/Boost Aiso的websocket 客户端/服务器库。
* [PcapPlusPlus](https://github.com/seladb/PcapPlusPlus): 跨平台的C++网络嗅探以及数据包解析和构建的框架。
* [ZeroMQ](http://zeromq.org/): 高速，模块化的异步通信库。

**PDF**

解析和处理PDF文档的库。

* [libharu](https://github.com/libharu/libharu): 免费、跨平台、开源的软件库，用于生成PDF。
* [litePDF](https://litepdf.sourceforge.io/): 创建和编辑PDF文档的库，它通过设备上下文使用GDI功能来绘制页面内容。
* [MuPDF](https://mupdf.com/): 轻量级的PDF、XPS和电子书查看器。
* [PoDoFo](http://podofo.sourceforge.net/): 使用PDF文件格式的库。
* [Poppler](https://poppler.freedesktop.org/): 基于xpdf-3.0代码库的开源、多后端的PDF渲染库。
* [Xpdf](https://www.xpdfreader.com/): Xpdf是一款免费的PDF查看器和工具包，包括文本提取器、图像转换器、HTML转换器等。
* [DynaPDF](http://www.dynaforms.com/): 易于使用的PDF生成库。

**物理学**

动力学仿真引擎

* [Box2D](https://code.google.com/p/box2d/): 2D的游戏物理引擎。
* [Bullet](https://github.com/bulletphysics/bullet3): 3D的游戏物理引擎。
* [Chipmunk](https://github.com/slembcke/Chipmunk2D): 快速、轻量级的2D游戏物理库。
* [LiquidFun](https://github.com/google/liquidfun): 2D的游戏物理引擎。
* [Newton Dynamics](https://github.com/MADEAPPS/newton-dynamics): 用于物理环境实时仿真的集成解决方案。
* [ODE](http://www.ode.org/): 开放动力学引擎-开源，高性能库，模拟刚体动力学。
* [ofxBox2d](https://github.com/vanderlin/ofxBox2d): Box2D开源框架包装器。
* [PhysX](https://github.com/NVIDIAGameWorks/PhysX-3.4): Nvidia开发的开源的实时物理引擎中间件SDK，作为Nvidia GameWorks软件套件的一部分。
* [Project Chrono](https://github.com/projectchrono/chrono): 开源的多物理场仿真引擎。
* [Quantum++](https://github.com/vsoftco/qpp): 现代的C++ 11量子计算库。
* [Simbody](https://github.com/simbody/simbody): 高性能C++多体动力学/物理库，模拟关节生物力学和机械系统，像车辆，机器人和人体骨骼。
* [SOFA](https://github.com/sofa-framework/sofa): SOFA是一个针对实时仿真的开源框架，其着重于医学仿真。

**映射**

* [Better Enums](https://github.com/aantron/better-enums): 映射的枚举类型（字符串、迭代）。单个头文件。
* [clReflect](https://bitbucket.org/dwilliamson/clreflect): 使用clang的C++映射。
* [CPFG](https://github.com/cpgf/cpgf): 用于反射、回调和脚本绑定的C++ 03库。
* [CPP-Reflection](https://github.com/AustinBrunkhorst/CPP-Reflection): 使用clang的C++映射。
* [Magic Enum](https://github.com/Neargye/magic_enum): 只包含头文件的的C++ 17库，为枚举类型（字符串、迭代）提供静态映射，可与任何枚举类型一起使用，无需任何宏或样板代码。
* [magic\_get](https://github.com/apolukhin/magic_get): 类似std: : tuple的方法，用于用户定义类型，无需任何宏或样板代码。
* [meta](https://github.com/skypjack/meta): C++的非侵入性且无宏的运行时映射系统，只包含头文件。
* [Nameof](https://github.com/Neargye/nameof): 只包含头文件的的C++ 17库，提供名称宏和函数以获取变量、类型、函数、宏和枚举的简单名称。
* [Ponder](https://github.com/billyquith/ponder): C++11的映射库。
* [RTTR](https://github.com/rttrorg/rttr): C++11的映射库。
* [visit\_struct](https://github.com/cbeck88/visit_struct): C++中用于结构域映射的微型库。

**正则表达式**

* [CppVerbalExpressions](https://github.com/VerbalExpressions/CppVerbalExpressions): C++正则表达式专家。
* [CTRE](https://github.com/hanickadot/compile-time-regular-expressions): 编译时PCRE（几乎）兼容的正则表达式匹配器。
* [Hyperscan](https://github.com/intel/hyperscan): Hyperscan是Intel的高性能、多种正则表达式匹配库，提供大量正则表达式（多达数万个）的同时匹配，通常在DPI库堆栈中使用。
* [Oniguruma](https://github.com/kkos/oniguruma): 现代且灵活的正则表达式库，支持多种字符编码。
* [PCRE](http://pcre.org/): 受Perl中正则表达式功能启发的正则表达式C库。
* [PIRE](https://github.com/yandex/pire): Yandex的Perl不兼容正则表达式库，可以非常快（超过400 MB / s）。
* [RE2](https://github.com/google/re2): 使用自动机理论、通过有限状态机进行的正则表达式软件库。
* [SLRE](https://github.com/cesanta/slre): 适用于C / C++的超轻正则表达式引擎。
* [sregex](https://github.com/openresty/sregex): 基于NFA / DFA的Perl兼容的正则表达式引擎库，不可回溯，可与大型数据流进行匹配。

**机器人学**

* [MOOS-IvP](http://moos-ivp.org/): 一组开源C++模块，提供机器人平台的自主权，尤其是自主的海洋车辆。
* [MRPT](http://www.mrpt.org/): 移动机器人编程工具包。
* [PCL](https://github.com/PointCloudLibrary/pcl): 点云库是一个独立的，大规模的开放项目，用于2D/3D图像和点云处理。
* [Robotics Library (RL)](http://www.roboticslibrary.org/): 一个独立的C++库，包括机器人动力学，运动规划和控制。
* [RobWork](http://www.robwork.dk/jrobwork/): 一组C++库的集合，用于机器人系统的仿真和控制。
* [ROS](http://wiki.ros.org/): 机器人操作系统，提供了一些库和工具帮助软件开发人员创建机器人应用程序。

**科学计算**

* [AMGCL](https://github.com/ddemidov/amgcl): 只包含头文件的C++库，用于解决具有代数多重网格的大型稀疏线性系统。
* [FFTW](http://www.fftw.org/): 用一维或者多维计算DFT的C语言库。
* [GSL](http://www.gnu.org/software/gsl/): GNU科学库。
* [TileDB](https://github.com/TileDB-Inc/TileDB): 快速密集和稀疏多维数组的DBMS。
* [Trilinos](https://github.com/trilinos/Trilinos): 高性能的PDE解算器。
* [Torch](https://github.com/torch/torch7): 科学的计算框架，广泛支持将GPU放在首位的机器学习算法。

**脚本**

* [AngelScript](https://www.angelcode.com/angelscript/): AngelScript是一种面向游戏的解释/编译脚本语言。
* [Boost.Python](http://www.boost.org/doc/libs/1_65_1/libs/python/doc/html/index.html): 可实现C++和Python编程语言之间的无缝互操作性的C++库。
* [cppimport](https://github.com/tbenthompson/cppimport): 直接从Python导入C++文件！
* [CppSharp](https://github.com/mono/CppSharp): 用于将C / C++ API融合到高级语言中的工具和库。
* [ChaiScript](https://github.com/ChaiScript/ChaiScript/): 用于C++的易于使用的嵌入式脚本语言。
* [ctypes.sh](https://github.com/taviso/ctypes.sh): bash的外部函数接口。
* [Cython](https://github.com/cython/cython): Cython是用于Python编程语言和扩展的Cython编程语言（基于Pyrex）的优化静态编译器。它使得为Python编写C扩展就像Python本身一样容易。
* [djinni](https://github.com/dropbox/djinni): 生成跨语言类型声明和接口绑定的工具。
* [Duktape](https://github.com/svaarala/duktape): 具有紧凑脚本的嵌入式Javascript引擎。
* [JavaCpp](https://github.com/bytedeco/javacpp): Java和本机C++之间缺少的桥梁。
* [libffi](https://github.com/libffi/libffi): 可移植的外部功能接口库。
* [Lua](http://www.lua.org/): 用于配置文件和基本应用程序脚本的小型快速脚本引擎。
* [LuaBridge](https://github.com/vinniefalco/LuaBridge): 轻量级、无依赖的库，用于将Lua绑定到C++。
* [luacxx](https://github.com/dafrito/luacxx): 用于创建Lua绑定的C++ 11 API。
* [nbind](https://github.com/charto/nbind): 神奇的头文件，使您的C++库可从JavaScript进行访问。
* [PHP-CPP](https://github.com/CopernicaMarketingSoftware/PHP-CPP): 使用C++构建PHP扩展的库。
* [pybind11](https://github.com/pybind/pybind11): C++ 11和Python之间的无缝可操作性。
* [SIP](https://riverbankcomputing.com/software/sip/intro): 用于Python v2和v3的C或C++绑定生成器。
* [sol2](https://github.com/ThePhD/sol2): 具有高级功能和一流性能的C++ <-> Lua API封装器。
* [SWIG](http://www.swig.org/): 一个可以让你的C++代码链接到JavaScript，Perl，PHP，Python，Tcl和Ruby的包装器/接口生成器。
* [V7](https://github.com/cesanta/v7): 嵌入式的JavaScript 引擎。
* [V8](http://code.google.com/p/v8/): 谷歌的快速JavaScript引擎，可以被嵌入到任何C++应用程序中。
* [ChakraCore](https://github.com/Microsoft/ChakraCore): Microsoft的JavaScript引擎，可以嵌入到nodejs中。

**序列化**

* [Bitsery](https://github.com/fraillt/bitsery): 只包含头文件的C++二进制序列化库。
* [Bond](https://github.com/Microsoft/bond): 用于处理模式化数据的开源、跨平台框架。
* [Boost.Serialization](https://www.boost.org/doc/libs/master/libs/serialization/doc/index.html): Boost序列化库。
* [Cap'n Proto](http://kentonv.github.io/capnproto/): 快速数据交换格式和RPC系统。
* [cereal](https://github.com/USCiLab/cereal): C++11 序列化库。
* [cppcodec](https://github.com/tplgy/cppcodec): 只包含头文件的C++ 11库，具有一致、灵活的API来编码/解码base64，base32和hex。
* [FlatBuffers](https://github.com/google/flatbuffers): 内存高效的序列化库。
* [MessagePack](https://github.com/msgpack/msgpack-c): C/C++的高效二进制序列化库，例如 JSON。
* [mrpt-serialization](https://github.com/mrpt/mrpt/): 已版本控制的二进制或文本格式的序列化。
* [nanopb](https://github.com/nanopb/nanopb): ANSI C中的小型代码大小的协议缓冲区实现。
* [protobuf](http://code.google.com/p/protobuf/): 协议缓冲，谷歌的数据交换格式。
* [protobuf-c](https://github.com/protobuf-c/protobuf-c): C语言的协议缓冲实现。
* [SimpleBinaryEncoding](https://github.com/real-logic/simple-binary-encoding): 用于低延迟应用程序的对二进制格式的应用程序信息的编码和解码。
* [upb](https://github.com/protocolbuffers/upb): C中的小型静态库实现。
* [YAS](https://github.com/niXman/yas): 非常快的序列化库，支持二进制/文本/ JSON格式。

**排序**

* [pdqsort](https://github.com/orlp/pdqsort): 破坏模式的快速排序。
* [Timsort](https://github.com/gfx/cpp-TimSort): 模板化的稳定排序功能，其性能优于基于快速排序的算法（包括std: : sort），用于反向或半排序数据。

**视频**

* [libvpx](http://www.webmproject.org/code/): VP8/VP9编码解码SDK。
* [FFmpeg](https://www.ffmpeg.org/): 一个完整的，跨平台的解决方案，用于记录，转换视频和音频流。
* [libde265](https://github.com/strukturag/libde265): 开放的h.265视频编解码器的实现。
* [x265](https://bitbucket.org/multicoreware/x265/wiki/Home): 开放的H.265视频编解码器实现。
* [OpenH264](https://github.com/cisco/openh264): 开源H.364 编解码器。
* [Theora](http://www.theora.org/): 免费开源的视频压缩格式。
* [Vireo](https://github.com/twitter/vireo/): Twitter的轻量级、多功能的视频处理库。

**虚拟机**

* [CarpVM](https://github.com/tekknolagi/carp): C中有趣的VM，让我们一起来看看这个。
* [MicroPython](https://github.com/micropython/micropython): 旨在实现单片机上Python3.x的实现。
* [TinyVM](https://github.com/jakogut/tinyvm): 用纯粹的ANSI C编写的小型，快速，轻量级的虚拟机。

**Web应用框架**

* [Civetweb](https://github.com/bel2125/civetweb): 提供易于使用，强大的，C/C++嵌入式Web服务器，带有可选的CGI，SSL和Lua支持。
* [C++ REST SDK](https://github.com/Microsoft/cpprestsdk): 使用现代异步C++ API设计、以本机代码进行基于云的客户端-服务器通信的Microsoft项目。
* [CppCMS](http://cppcms.com/): 免费高性能的Web开发框架（不是 CMS）.。
* [Crow](https://github.com/ipkn/crow): 一个C++微型web框架（灵感来自于Python Flask）。
* [Cutelyst](https://github.com/cutelyst/cutelyst): 在Qt上构建的C++ Web框架，使用Catalyst（Perl）框架的简单方法。
* [Drogon](https://github.com/an-tao/drogon): 基于C++ 14/17的高性能HTTP应用程序框架。
* [facil.io](https://github.com/boazsegev/facil.io): 事件型、高性能的C Web框架，支持HTTP、WebSocket、SSE等。
* [Kore](https://kore.io/): 使用C语言开发的用于web应用程序的超快速和灵活的web服务器/框架。
* [libOnion](http://www.coralbits.com/libonion/): 轻量级的库，帮助你使用C编程语言创建web服务器。
* [lwan](https://github.com/lpereira/lwan): 实验性、可扩展的高性能HTTP服务器。
* [oat++](https://github.com/oatpp/oatpp): 轻量、零依赖的框架，可创建高性能的Web服务。
* [Pistache](http://pistache.io/): Pistache是一个用纯C++ 11编写的C++ REST框架，没有外部依赖性。
* [QDjango](https://github.com/jlaine/qdjango/): 使用C++编写的，基于Qt库的web框架，试图效仿Django API，因此得此名。
* [TreeFrog Framework](https://github.com/treefrogframework/treefrog-framework): 基于C++和Qt的高速、全栈的Web应用程序框架，支持HTTP和WebSocket协议（带有O / R映射）。
* [Wt](http://www.webtoolkit.eu/wt): 开发Web应用的C++库。

**XML**

XML就是个垃圾，xml的解析很烦人，对于计算机它也是个灾难。这种糟糕的东西完全没有存在的理由了。-Linus Torvalds

* [Boost.PropertyTree](http://www.boost.org/doc/libs/1_55_0/doc/html/property_tree.html): 属性树解析器/生成器，可用于解析XML / JSON / INI / Info文件。
* [Expat](http://www.libexpat.org/): 用C语言编写的xml解析库。
* [Libxml2](http://xmlsoft.org/): Gnome的xml C解析器和工具包。
* [libxml++](http://libxmlplusplus.sourceforge.net/): C++的xml解析器。
* [Mini-XML](https://github.com/michaelrsweet/mxml): 用ANSI C编写的小型XML解析库。
* [PugiXML](http://pugixml.org/): 用于C++的，支持XPath的轻量级，简单快速的XML解析器。
* [RapidXml](http://rapidxml.sourceforge.net/): 试图创建最快速的XML解析器，同时保持易用性，可移植性和合理的W3C兼容性。
* [TinyXML](http://sourceforge.net/projects/tinyxml/): 简单小型的C++XML解析器，可以很容易地集成到其它项目中。
* [TinyXML2](https://github.com/leethomason/tinyxml2): 简单快速的C++CML解析器，可以很容易集成到其它项目中。
* [TinyXML++](https://code.google.com/p/ticpp/): TinyXML的一个全新的接口，使用了C++的许多许多优势，模板，异常和更好的异常处理。
* [Xerces-C++](http://xerces.apache.org/xerces-c/): 用可移植的C++的子集编写的XML验证解析器。

**多项混杂**

一些有用的库或者工具，但是不适合上面的分类，或者还没有分类。

* [access\_profiler](https://github.com/arvidn/access_profiler): 用于计算对c ++程序中成员变量的访问的工具。
* [American fuzzy lop](http://lcamtuf.coredump.cx/afl/): a.k.a. afl-fuzz-疯狂的模糊测试工具，可在给定时间和最少示例输入的情况下自动发现错误。
* [Better String](http://bstring.sourceforge.net/): C字符串库的替代方法，它功能更强大，并且没有缓冲区溢出超限问题。还包括C++封装器。
* [Boost.Signals](http://www.boost.org/doc/libs/1_56_0/doc/html/signals.html): 托管信号和插槽系统的实现。
* [CPPItertools](https://github.com/ryanhaining/cppitertools): 受Python内建函数和itertools库启发的基于范围的循环附加组件。
* [DynaMix](https://github.com/iboB/dynamix): 允许您在运行时组成和修改对象的库。
* [{fmt}](https://github.com/fmtlib/fmt): 小型、安全、快速的C++格式化库。
* [FastFormat](http://www.fastformat.org/): 受log4j和Pantheios启发的快速、安全的C++格式化。
* [C++ Format](https://github.com/cppformat/cppformat): C++的小型，安全和快速格式化库。
* [casacore](https://code.google.com/p/casacore/): 从aips++ 派生的一系列C++核心库。
* [QtVerbalExpressions](https://github.com/VerbalExpressions/QtVerbalExpressions): 该Qt库基于C++ VerbalExpressions库。
* [cxx-prettyprint](https://github.com/louisdx/cxx-prettyprint): 用于C++容器的打印库。
* [icecream-cpp](https://github.com/renatoGarcia/icecream-cpp): 用于调试目的的小型打印库。
* [gcc-poison](https://github.com/leafsr/gcc-poison): 帮助开发人员禁止应用程序中的不安全的C/C++函数的简单的头文件。
* [Kangaru](https://github.com/gracicot/kangaru): 用于C++ 11和C++ 14的依赖注入容器。
* [Klib](https://github.com/attractivechaos/klib): 通用算法和数据结构的小型、轻量级实现。
* [libsigc++](http://libsigc.sourceforge.net/): 用于标准C++的类型安全的回调系统。
* [googlemock](http://code.google.com/p/googlemock/): 编写和使用C++模拟类的库。
* [HTTP Parser](https://github.com/joyent/http-parser): C的http请求/响应解析器。
* [libcpuid](https://github.com/anrieff/libcpuid): 用于x86 CPU检测盒特征提取的小型C库。
* [libevil](https://github.com/avati/libevil): 许可证管理器
* [libnih](https://github.com/keybuk/libnih): C函数和结构的轻量级库。
* [libusb](http://www.libusb.org/): 允许移动访问USB设备的通用USB库。
* [Mach7](https://github.com/solodon4/Mach7): C++的模式匹配库。
* [mio](https://github.com/mandreyel/mio): 用于内存映射文件IO的跨平台C++ 11库，只包含头文件。
* [PEGTL](https://github.com/taocpp/PEGTL): 解析表达语法模板库。
* [PCRE](http://pcre.org/): 正则表达式C库，灵感来自于Perl中正则表达式的功能。
* [Remote Call Framework](http://www.deltavsoft.com/): C++的进程间通信框架。
* [Scintilla](http://scintilla.org/): 开源的代码编辑控件。
* [Serial Communication Library](https://github.com/wjwwood/serial): C++语言编写的跨平台，串口库。
* [SDS](https://github.com/antirez/sds): C的简单动态字符串库。
* [semver.c](https://github.com/h2non/semver.c): semver解析器，在ANSI C中呈现。
* [Serial Communication Library](https://github.com/wjwwood/serial): 跨平台，用C++编写的串行端口库。
* [sigslot](http://sigslot.sourceforge.net/): C++信号/插槽库。
* [SimpleSignal](https://github.com/larspensjo/SimpleSignal): 高性能C++ 11信号。
* [Stage](https://github.com/rtv/Stage): 移动机器人模拟器。
* [stdman](https://github.com/jeaye/stdman): 格式化的C++ 11/14 stdlib手册页（来自cppreference）。
* [stb](https://github.com/nothings/stb): 一系列用于C / C++的单文件库。
* [StrTk](http://www.partow.net/programming/strtk/index.html): 由高性能字符串处理例程组成的C++库。
* [Tulip Indicators](https://tulipindicators.org/): 超过100种财务技术分析指标的C库。
* [ub-canaries](https://github.com/regehr/ub-canaries): 一组C / C++程序，它们试图使编译器开发未定义的行为。
* [value-category-cheatsheet](https://github.com/jeaye/value-category-cheatsheet): 用于lvalue、rvalue等的PDF快捷键管理。
* [VarTypes](https://code.google.com/p/vartypes/): C++/Qt4功能丰富，面向对象的管理变量的框架。
* [ZBar](http://zbar.sourceforge.net/): ‘条形码扫描器’库，可以扫描照片，图片和视频流中的条形码，并返回结果。
* [ZXing](https://github.com/zxing/zxing/): 用Java实现的开源、多格式1D / 2D条码图像处理库，带有其他语言的端口。
* [pprint](https://github.com/p-ranav/pprint): 适用于现代C++的Pretty Printer。

**软件**

用于创建开发环境的软件

**编译器**

C/C++编译器列表

* [8cc](https://github.com/rui314/8cc): 小型C编译器。
* [c](https://github.com/ryanmjacobs/c): 一次编译并执行C“脚本”！
* [Clang](http://clang.llvm.org/): LLVM的C编译器，支持C++ 11/14 / 1z C11，由LLVM团队开发。
* [GCC](https://gcc.gnu.org/): GNU编译器集合，支持C++ 11/14 / 1z C11和OpenMP。
* [PCC](http://pcc.ludd.ltu.se/): 非常老的C编译器，支持C99。
* [Intel C++ Compiler](https://software.intel.com/en-us/c-compilers): 由英特尔公司开发。
* [LLVM](http://llvm.org/): 模块化和可重用编译器和工具链技术的集合。
* [Microsoft Visual C++](http://msdn.microsoft.com/en-us/vstudio/hh386302.aspx): MSVC，由微软公司开发。
* [Open WatCom](http://www.openwatcom.org/index.php/Main_Page): Watcom，C，C++和Fortran交叉编译器和工具。
* [Oracle Solaris Studio](http://www.oracle.com/technetwork/server-storage/solarisstudio/overview/index.html): 适用于SPARC和x86的C、C++和Fortran编译器，支持C++ 11 ，在Linux和Solaris上可用。
* [TCC](http://bellard.org/tcc/): 轻量级的C语言编译器。

**在线编译器**

在线C/C++编译器列表

* [codepad](http://codepad.org/): 在线编译器/解释器，一个简单的协作工具
* [coliru](http://coliru.stacked-crooked.com/): 在线编译器/ shell，支持各种C++编译器。
* [Compiler Explorer](http://gcc.godbolt.org/): 具有编译输出的交互式编译器。
* [CompileOnline](http://www.tutorialspoint.com/codingground.htm): 在Linux上在线编译和执行C++。
* [Ideone](http://ideone.com/): 在线编译器和调试工具，可编译源代码并以60多种编程语言在线执行。
* [repl.it](https://repl.it/): 针对教育者、学习者和开发人员的功能强大而简单的工具和平台。
* [Rextester](http://rextester.com/runcode): 在线编译器，提供几个编译器（Clang，GCC，MSVC）和几个编辑器。
* [Try It Online](https://tio.run/): TIO是一系列在线解释器，可用于越来越多的实用和休闲的编程语言。
* [Wandbox](http://melpon.org/wandbox/): 具有Boost的在线Clang / GCC编译器。
* [paiza.io](https://paiza.io/en): 具有多个文件支持功能、Github（gist）集成和协作编辑的在线C / C++编译器。

**调试器**

C/C++调试器列表

* [Comparison of debuggers](http://en.wikipedia.org/wiki/Comparison_of_debuggers): 来自维基百科的调试器列表。
* [GDB](https://www.gnu.org/software/gdb): GNU调试器。
* [LLDB](http://lldb.llvm.org/): LLDB调试器。
* [Metashell](https://metashell.readthedocs.org/): 交互式模板元程序设计，其中包括MDB metadebugger。
* [Valgrind](http://valgrind.org/): 内存调试，内存泄露检测，性能分析工具。
* [x64dbg](http://x64dbg.com/): Windows的开源x64 / x32调试器。

**集成开发环境**

C/C++集成开发环境列表

* [Anjuta DevStudio](http://anjuta.org/): GNOME IDE。
* [AppCode](http://www.jetbrains.com/objc/): 构建与JetBrains’ IntelliJ IDEA 平台上的用于Objective-C，C,C++，Java和Java开发的集成开发环境。
* [Cevelop](https://www.cevelop.com/): 基于Eclipse CDT的跨平台C和C++ IDE，带有其他插件。
* [CLion](http://www.jetbrains.com/clion/): 来自JetBrains的跨平台的C/C++的集成开发环境。
* [Code::Blocks](http://www.codeblocks.org/): 免费C，C++和Fortran的集成开发环境。
* [CodeLite](http://codelite.org/): 另一个跨平台的免费的C/C++集成开发环境。
* [color\_coded](https://github.com/jeaye/color_coded): 用于基于libclang的vim插件。
* [Dev-C++](http://sourceforge.net/projects/orwelldevcpp/): 可移植的C/C++/C++11集成开发环境。
* [Eclipse CDT](http://www.eclipse.org/cdt/): 基于Eclipse平台的功能齐全的C和C++集成开发环境。
* [Geany](http://www.geany.org/): 轻量级的快速，跨平台的集成开发环境。
* [IBM VisualAge](http://www-03.ibm.com/software/products/en/visgen): 来自IBM的家庭计算机集成开发环境。
* [Irony-mode](https://github.com/Sarcasm/irony-mode): 由libclang驱动的用于Emacs的C/C++微模式。
* [KDevelop](https://www.kdevelop.org/): 免费开源集成开发环境。
* [Microsoft Visual Studio](http://www.visualstudio.com/): 来自微软的集成开发环境。
* [Microsoft Visual Studio Code](https://code.visualstudio.com/): Microsoft提供的开源IDE。
* [NetBeans](https://netbeans.org/): 主要用于Java开发的的集成开发环境，也支持其他语言，尤其是PHP，C/C++和HTML5。
* [Qt Creator](http://qt-project.org/): 跨平台的C++，Javascript和QML集成开发环境，也是Qt SDK的一部分。
* [rtags](https://github.com/Andersbakken/rtags): C/C++的客户端服务器索引，用于 跟基于clang的emacs的集成。
* [Xcode](https://developer.apple.com/xcode/): 由苹果公司开发。
* [YouCompleteMe](https://valloric.github.io/YouCompleteMe/): 一个用于Vim的根据你敲的代码快速模糊搜索并进行代码补全的引擎。
* [cquery](https://github.com/cquery-project/cquery/): 用于vscode，emacs，vim等的C++代码完成引擎。

**构建系统**

* [Bazel](https://bazel.build/): Google提供的多语言、快速且可扩展的构建系统。
* [Bear](https://github.com/rizsotto/Bear): 用于为clang工具生成编译数据库的工具。
* [Buck](https://github.com/facebook/buck): 用Java编写的快速构建系统，可鼓励在多种平台和语言（包括在Facebook开发和使用的C++）上创建小型、可重用模块。
* [clib](https://github.com/clibs/clib): C语言的软件包管理器。
* [Biicode](https://www.biicode.com/): 基于文件的简单依赖管理器。
* [CMake](http://www.cmake.org/): 跨平台的免费开源软件用于管理软件使用独立编译的方法进行构建的过程。
* [Cget](https://github.com/pfultz2/cget): Cmake软件包检索。
* [Conan](https://conan.io/): C / C++程序包管理器，开源的。
* [CPM](https://github.com/iauns/cpm): 基于CMake和Git的C++包管理器。
* [FASTBuild](http://www.fastbuild.org/docs/home.html): 高性能，开源的构建系统，支持高度可扩展性的编译，缓冲和网络分布。
* [Hunter](https://www.github.com/ruslo/hunter): CMake驱动的C++跨平台软件包管理器。
* [MesonBuild](http://mesonbuild.com/): 开源构建系统，它意味着不仅要非常快，而且更重要的是要尽可能地方便用户使用。
* [Ninja](http://martine.github.io/ninja/): 专注于速度的小型构建系统。
* [Scons](http://www.scons.org/): 使用Python scipt 配置的软件构建工具。
* [Sconsolidator](http://www.sconsolidator.com/): 用于Eclipse CDT的Scons构建系统集成。
* [Spack](https://spack.io/): 灵活的软件包管理器，支持多个版本、配置、平台和编译器。
* [SW](https://software-network.org/): 跨平台的C++（和其他语言）构建系统和程序包管理器，其中包含许多可用程序包。
* [tundra](https://github.com/deplinenoise/tundra): 高性能的代码构建系统，甚至对于非常大型的软件项目，也能提供最好的增量构建次数。
* [tup](http://gittup.org/tup/): 基于文件的构建系统，用于后台监控变化的文件。
* [Premake](http://premake.github.io/): 配置有Lua脚本的工具，可以在Windows、Mac OS X和Linux上为Visual Studio、GNU Make、Xcode、Code::Blocks等生成项目文件。
* [Vcpkg](https://github.com/microsoft/vcpkg): Windows，Linux和MacOS的C++库管理器。
* [waf](https://github.com/waf-project/waf): 用于配置、编译和安装应用程序的基于Python的框架。
* [XMake](https://xmake.io/): 基于Lua的C / C++的跨平台构建实用程序。

**静态代码分析**

提高质量，减少瑕疵的代码分析工具列表

* [Cppcheck](http://cppcheck.sourceforge.net/): 静态C/C++代码分析工具
* [CppDepend](https://www.cppdepend.com/): 通过分析和可视化代码依赖关系，定义设计规则，进行影响分析以及比较不同版本的代码，简化了对复杂C / C++代码库的管理。
* [cpplint](https://github.com/cpplint/cpplint): 遵循Google C++样式指南的C++样式检查器。
* [PVS-Studio](http://www.viva64.com/en/pvs-studio/): 用C、C++和C＃编写的在程序源代码中进行bug检测的工具。
* [cpp-dependencies](https://github.com/tomtom-international/cpp-dependencies): 检查C++ #include依赖关系的工具（以.dot格式创建的依赖关系图）。
* [include-what-you-use](https://code.google.com/p/include-what-you-use/): 使用clang进行代码分析的工具，可以#include在C和C++文件中。
* [Infer](https://github.com/facebook/infer): 用于Java、C和Objective-C的静态分析器。
* [OCLint](http://oclint.org/): 用于C、C++和Objective-C的静态源代码分析工具，用于提高质量，减少瑕疵。
* [Clang Static Analyzer](http://clang-analyzer.llvm.org/index.html): 查找C、C++和Objective-C程序bug的源代码分析工具。
* [Linticator](http://linticator.com/): Pc-/FlexeLint的Eclipse CDT集成。
* [IKOS](https://github.com/NASA-SW-VnV/ikos): 基于抽象解释理论的C / C++静态分析器。
* [List of tools for static code analysis](http://en.wikipedia.org/wiki/List_of_tools_for_static_code_analysis#C.2FC.2B.2B): 来自维基百科的静态代码分析工具列表。

**编码样式工具**

* [Artistic Style](http://astyle.sourceforge.net/): 格式化C / C++ / C＃/ Obj-C / Java代码的工具。也称为astyle。
* [ClangFormat](http://clang.llvm.org/docs/ClangFormat.html): 格式化C / C++ / Obj-C代码的工具。
* [Clang-Tidy](http://clang.llvm.org/extra/clang-tidy.html): 基于Clang的C++ linter工具。
* [EditorConfig](https://editorconfig.org/): EditorConfig帮助在不同的编辑器和IDE之间维护一致的编码样式
* [Uncrustify](https://github.com/uncrustify/uncrustify): 代码美化器。

**About**

C++ 资源大全中文版，标准库、Web应用框架、人工智能、数据库、图片处理、机器学习、日志、代码分析等。由「开源前哨」和「CPP开发者」微信公号团队维护更新。

**Topics**

[cpp](https://github.com/topics/cpp)

**Resources**

[Readme](https://github.com/jobbole/awesome-cpp-cn#readme)

**Stars**

[**6.3k** stars](https://github.com/jobbole/awesome-cpp-cn/stargazers)

**Watchers**

[**422** watching](https://github.com/jobbole/awesome-cpp-cn/watchers)

**Forks**

[**1.7k** forks](https://github.com/jobbole/awesome-cpp-cn/network/members)

[**Releases**](https://github.com/jobbole/awesome-cpp-cn/releases)

No releases published

[**Packages**](https://github.com/orgs/jobbole/packages?repo_name=awesome-cpp-cn)

No packages published

[**Contributors**9](https://github.com/jobbole/awesome-cpp-cn/graphs/contributors)

* [](https://github.com/tangyouhua)
* [](https://github.com/huanglimin)
* [](https://github.com/ShenMian)
* [](https://github.com/ZZMarquis)
* [](https://github.com/hanxiaomax)
* [](https://github.com/jzaicn)
* [](https://github.com/cxwx)
* [](https://github.com/idealvin)
* [](https://github.com/Futrime)
* © 2022 GitHub, Inc.
* [Terms](https://docs.github.com/en/github/site-policy/github-terms-of-service)
* [Privacy](https://docs.github.com/en/github/site-policy/github-privacy-statement)
* [Security](https://github.com/security)
* [Status](https://www.githubstatus.com/)
* [Docs](https://docs.github.com/)
* [Contact GitHub](https://support.github.com/?tags=dotcom-footer)
* [Pricing](https://github.com/pricing)
* [API](https://docs.github.com/)
* [Training](https://services.github.com/)
* [Blog](https://github.blog/)
* [About](https://github.com/about)