Software Hardware List

Chapter number	Software required (With version)	Download links to the software	Hardware specifications	OS required
All	GCC 8.0	`https://gcc.gnu.org/`	2 GB of RAM and 20GB of disk space. A Virtual Machine with this characteristics should suffice.	Linux Mac OS
All	Clang 7.0	`http://clang.llvm.org/`	2 GB of RAM and 20GB of disk space. A Virtual Machine with this characteristics should suffice.	Linux Mac OS
All	Visual Studio Community 15.3 or newer	`https://www.visualstudio.com/vs/`	2 GB of RAM and 20GB of disk space. A Virtual Machine with this characteristics should suffice.	Windows (7, 8.x, or 10)
5	date	`https://github.com/HowardHinnant/date`		Cross-platform
7	stduuid	`https://github.com/mariusbancila/stduuid`		Cross-platform
9	pugixml	`https://pugixml.org/`		Cross-platform
9	nlohmann/json	`https://github.com/nlohmann/json`		Cross-platform
9	pdfwriter	`https://github.com/galkahana/PDF-Writer`		Cross-platform
10	ZipLib	`https://bitbucket.org/wbenny/ziplib.git`		Cross-platform

10	PNGWriter	`https://github.com/pngwriter/pngwriter`		Cross-platform
10	SQLite	`https://www.sqlite.org/`		Cross-platform
10	sqlite_modern_cpp	`https://github.com/SqliteModernCpp/sqlite_modern_cpp`		Cross-platform
11	Crypto++	`https://www.cryptopp.com/`		Cross-platform
12	Asio	`https://think-async.com/`		Cross-platform
12	curl	`https://curl.haxx.se/`		Cross-platform
12	curlcpp	`https://github.com/JosephP91/curlcpp`		Cross-platform
13	openssl	`https://www.openssl.org/`		Linux Mac OS
14	boost	`https://www.boost.org/`		Cross-platform

The code presented in this book uses language and library features introduced in C++ 11/14/17. Therefore, you need a compiler that supports C++17 entirely, or at least partially (`std::string_view`, `std::optional`, the `filesystem` library are used throughout the book). You can use either Clang 6 or GCC 8 to run the code on Mac or Linux systems, and Visual C++ 15.6 for Windows.

Several proposed solutions are using the `filesystem` library. At the time of writing the book, this library is available with Visual C++ 15.3 and GCC 8, but not with Clang. If you are using Xcode on Mac, you will not be able to compile those solutions because Xcode does not support GCC. In this case, you can use *Boost.Filesystem*, which was the model for the standard filesystem library. Similarly, `std::optional` is used in several solutions and the Clang compiler distributed with Xcode does not support it. Again, you can use Boost.optional as a replacement. There are no changes required to the code when using these Boost libraries, except for the included headers and namespaces. Actually, the code provided with the book works with both *Boost.Filesystem* and *Boost.optional* on one hand, and the standard `filesystem` library and `std::optional` on the other hand. Please refer to the instructions for building the source code available both in the preface and source code archive.

All the major C++ compilers are available online and can be used to run many of the solutions provided in this book, with the exceptions of those requiring 3rd-party libraries. Note that new versions of the compilers are added to these online resources all the time. The compiler versions referred in the table below are from May 2018.

Chapter version	Compiler	Weblink
All	GCC (many versions including development trunk) Clang (many versions including development trunk)	`https://wandbox.org/`
All	GCC (many versions including 7.0) Clang (many versions including 6.0.0) Intel C++ Compiler (several versions, latest 18) Visual C++ 2017 (several versions)	`https://godbolt.org/`

Apart from the standard library, many 3rd-party libraries are used throughout the book. All the libraries used in the book are open-sourced and cross-platform. Instructions for installing or building the library are available on each project's website. In some cases, additional information is provided in the book.