# Simple-Web-Server: a fast and flexible HTTP/1.1 C++ client and server library

**Ole Christian Eidheim**[1]

**1** Department of Computer Science, Norwegian University of Science and Technology

**DOI:** 10.21105/joss.01592

**Software**

- Review
- Repository
- Archive

**Submitted:** 18 July 2019
**Published:** 26 August 2019

## Summary

The programming language C++ is commonly used for resource intensive tasks. Simple-Web-Server is a library that can be utilized in C++ applications to implement web-resources or perform HTTP or HTTPS requests in a simple manner across OS platforms compared to using a networking library directly. Thus, Simple-Web-Server can be helpful for any research software written in C++ that needs to communicate with remote endpoints through HTTP or HTTPS.

The main features, apart from speed and ease of use, are flexibility and safety. The asynchronous I/O library Asio C++ Library (Kohlhoff, 2003) is used to implement networking and asynchronous event handling. The sending of outgoing messages has been made thread safe, and event handling in one or several threads is supported. The default event handling strategy is using one thread, commonly called event-loop, which makes accessing shared resources safe without using resource locking through for instance mutexes. Although, accessing shared resources in a multithreaded event-handling strategy can be made safer by utilizing the annotation offered in Clang Thread Safety Analysis (The Clang Team, 2007). In some cases, however, processing requests sequentially, in an event-loop scheme, can be faster than processing the requests in several threads where shared resources must be protected from simultaneous use.

An additional safety feature is stopping of asynchronous handlers when the associated client or server object has been destroyed. An atomic instruction based class, ScopeRunner, was implemented to achieve this since reader-writer locks proved more resource intensive for this specific task. In detail, a ScopeRunner object has an internal atomic counter that is increased when an asynchronous handler is run. At the end of the handler, the counter is decreased. When the destructor of a client or server object is called, the ScopeRunner object delays the destructor until its internal counter is 0, then sets the counter to a negative value. Finally, when the internal counter is negative, the handlers are returned from instead of potentially calling methods or using member variables of a destroyed client or server object.

Compared to using a low-level network library, specialized for a specific task, a slight performance overhead is expected when using the more generalized Simple-Web-Server library. The various utility and safety features, and code abstractions contribute to this overhead, but a good balance between safety, usability and speed is continuously sought during development of this library. Regular expressions can for instance be used to define which functions to be called for specific request paths. This can be convenient for the library user, but a more specific algorithm can be more efficient than using regular expressions.

The Asio C++ Library (Kohlhoff, 2003) is currently proposed to the C++ standard library (Wakely, 2017). If accepted in one of the future revisions of the C++ programming language, C++ applications can make use of a standardized event handling system. Until then, efforts are made to support old and new versions of the Asio C++ Library, as well as both the standalone and Boost variants of the library.

Simple-Web-Server is used in teaching at the Norwegian University of Science and Technology, and used in many external projects, for instance in the multi-purpose emulation framework MAME (MAMEDev, 1997). The library was also used in the senior thesis by Chung and Callin (Chung & Callin, 2017). Furthermore, one of the motivations for the Simple-Web-Server project was to create a HTTP/1.1 library that was relatively easy to modify and extend to suit a specific need, which could also be positive with regards to source code contributions to the project.

There are several alternatives to Simple-Web-Server. Most notably Boost.Beast (Falco, 2016), but this library is made for library authors and is thus harder to utilize in a C++ application. Additionally, Boost.Beast does not support standalone Asio. Another alternative is H2O (DeNA Co., Ltd., 2014) that supports several event handling systems, however, Asio is not yet supported. Both Boost.Beast, and to a lesser degree H2O, supports the WebSocket protocol (Fette & Melnikov, 2011). In the case of Simple-Web-Server, WebSocket is supported through a related external project named Simple-WebSocket-Server (Eidheim, 2014).

Based on Simple-Web-Server, a new C++ library supporting HTTP/2 is under development. HTTP/2 is very different from HTTP/1.1, but the experiences from developing Simple-Web-Server, and some its implementations, such as the ScopeRunner class, can be helpful when writing an HTTP/2 library.

## Acknowledgments

I would like to thank all those who have contributed to the Simple-Web-Server project.

## References

Chung, M., & Callin, J. (2017). Point cloud framework for rendering 3D models using Google Tango. Computer Science and Engineering Senior Theses, Santa Clara University; Santa Clara: Santa Clara University, 2017. Retrieved from https://scholarcommons.scu.edu/cseng_senior/84

DeNA Co., Ltd. (2014). H2O. Retrieved July 17, 2018, from https://github.com/h2o/h2o

Eidheim, O. C. (2014). Simple-websocket-server. Retrieved July 17, 2018, from https://gitlab.com/eidheim/Simple-WebSocket-Server

Falco, V. (2016). Boost.Beast. Retrieved July 17, 2018, from https://github.com/boostorg/beast

Fette, I., & Melnikov, A. (2011, December). The websocket protocol. RFC, RFC Editor; Internet Requests for Comments; RFC Editor. doi:10.17487/RFC6455

Kohlhoff, C. M. (2003). Asio c++ library. Retrieved July 17, 2018, from https://think-async.com/Asio/

MAMEDev. (1997). MAME. Retrieved July 17, 2018, from https://www.mamedev.org/

The Clang Team. (2007). Clang thread safety analysis. Retrieved July 17, 2018, from https://clang.llvm.org/docs/ThreadSafetyAnalysis.html

Wakely, J. (2017). Working draft, c++ extensions for networking. Retrieved July 17, 2018, from http://www.open-std.org/jtc1/sc22/wg21/docs/papers/2017/n4656.pdf