Libsocketpp

A tutorial to C++ streambased sockets and accompanying tools

Charlie Sale

This manual is for libsocketpp, 0.2

Published by GNU Manual

Table of Contents

1 Overview

Welcome to Libsocketpp, the C++ socket library based on the standard C++ I/O system. Because it is built on top of the standard C++ I/O system, sockets behave the same way as an `istream` or `ostream` would work. Lets go over what that means exactly.

By saying that libsocketpp is built on top of the C++ standard I/O system, I mean that it inherits standard classes included in the standard C++ library that are built for I/O functionality. Classes like `cout` and `cin` are built on the same systems. The specific classes that libsocketpp are built on are `streambuf`, which is a typedef of `basic_streambuf`, and `iostream`, which is a typedef of `basic_iostream`. Let's cover what these classes do, and then how they relate to libsocketpp.

Let's start with the class `iostream`. This class is whats known as a stream. A stream is a device that formats and transports data over a connection. This means that it physically reads and writes data between two places. The class `iostream` both reads and writes data via methods inherited from `istream` and `ostream`. As said earlier, a stream only moves data, which means that we are missing a part of the system: the buffer.

A buffer, embodied in the class `streambuf`, is the location in memory in which data is stored to retrieved for sending and recieving. When a stream writes data, it retrieves data from the the buffer and outputs it to the destination. Conversly, when a stream reads data, it places retrieved data into the buffer. Once this read data is placed in the buffer, said data can be returned to the user.

In order for a stream to use a buffer, a buffer is usually passed into it's constructor. In libsocketpp, both the buffer and stream are combined into one class for easy usage.

A question that you might have is "How does this I/O system relate to sockets¿' Well, the type of socket primarily used in this library is a Transmission Control Protocol (TCP) socket. A TCP socket is itself a stream. This means that data is transfered over a connection through blocking read and write commands. Because TCP is a stream, it fits perfectly into the C++ I/O system.

2 Tutorial

Let's now talk about how to use this library.

2.1 Acquring and Installing

2.1.1 Acquiring

2.1.2 Installing

Because this package follows standard GNU build proceedures, install with the following steps:

1. `$ cd libsocketpp-dir`
2. `$ ./configure --prefix=install-prefix`
3. `$ make`
4. `# make install`

2.2 Compiling

This library installs both a static and dynamic library. If you want to keep dependencies down or want to have a very portable program/library, copy the static library (`prefix/lib/libsocketpp.a`) into your project directory and compile it it by doing

`$ g++ file.cc libsocketpp.a -o executable`.

If you want to compile with the dynamic library, then compile with the flag `-lsocketpp`. This is assuming that your prefix is part of the standard library serach path of ld. If something goes wrong here, then try running `# ldconfig`.

2.3 Including

All libsocketpp headers are found in the directory `socketpp/` in your prefix include directory. From there, classes are broken down by specific function or protocol. Both the `Socket` and `Server` classes are found int the `tcp/` directory. Note that these subdirectories correspond with the classes namespace. This means the `Socket` is part of the namespace `tcp`.

For example:

```
#include <socketpp/tcp/socket.h>
#include <socketpp/tcp/server.h>

// so you don't have to type tcp:: everytime
using namespace tcp;

/* The rest of your project here */
```

2.4 Handling Errors

Errors in socketpp are fairly low level. Most functions return an integer value for testing if they do not already return some sort of other data type. Generally, a function will return a negative value (i.e -1). On success, most functions return `0` on success, so to test, you can write:

```
int ret = socket_function();
if (ret != 0){
   // handle error
}

// or even easier
int ret = socket_function();
if (!ret)
   // handle error
```

For more specific error handling, all standard C socket `errno` values are still set. There are no new `errno` values set by this library: they just recycle what would already be set by the standard system. Look at the errno documentation at the website ‘`http://www.virtsync.com/c-error-codes-include-errno`’. This contains the number and a brief description about each errno error value.

The constructors for `tcp::Socket` and `tcp::Server` both throw the `ctor_exe_t` exception when calling their constructors. The constructor does not inherently throw the exception, but if an error in encountered, it will abort and dump the core. Because of this, it is reconmended that all constructor calls are acompnaied with a try/catch block.

2.5 Constructors, Connecting, and Binding

Now, let’s get into the actual classes and methods of libsocketpp.

Let’s start with the `tcp::Socket` class. This class is the main class to use when making connections to servers.

The first thing to do when using a `tcp::Socket` is to instantiate an object and connect to a server. Here is an example of how to do that:

`test.cc`

```
#include <socketpp/tcp/socket.h>
#include <iostream>

using namespace tcp;

int main()
{


  Socket* sock;
  try {
```

```
    sock = new Socket();
  } catch (ctor_exe_t& e) {
    cerr << e.what() << endl;
  }

  sock->connects("127.0.0.1", 8888);

  sock->closes();

  return 0;
}
```

In this example, a simple socket object is created and connects to `localhost` on port 8888. Lets go through this simple program line by line:

1.

   ```
   #include <socketpp/tcp/socket.h>
   #include <iostream>

   using namespace tcp;
   ```

   These two lines include the `tcp::Socket` class and tell the processor to use the `tcp` namespace as a prefix to all classes and methods if necessary. The `using namespace tcp` is not necessary, but is easier to use. However, if you are working on a large project, it may be a good idea to not use the using clause in order to separate out classes. It's up to you.

2.

   ```
   int main()
   {
     ...

     return 0;
   }
   ```

   This is simply the main method, or the main entry point of the program. If you are a seasoned C developer, this should not be new to you. However, if you are not, then give it a google search.

3.

   ```
     Socket* sock;
     try {
       sock = new Socket();
     } catch (ctor_exe_t& e) {
       cerr << e.what() << endl;
     }
   ```

```
        sock->connects("127.0.0.1", 8888);

        sock->closes();
```

This is the meat of the program. Lets start with the `Socket sock;` line. This line is the constructor for the `tcp::Socket` class. Note that, because of the `using` statement, the `tcp::` is omitted. The constructor used in this program is the simple constructor; all it does is create on object, nothing more. There are multiple constructors, however. One variant of the constructor is `Socket(const char*, int)`. This constructor gives the socket data to connect to the server. **TODO: CHECK THE ACCURACY OF THE PREVIOUS STATEMENT**. The try catch block around the constructor catchs the chance of the constructor failing. There is another constructor, `Socket(const char*, int, int);`, where if the final integer is non-zero (true), then the socket will connect in the constructor. The final constructor is the `Socket(int);` constructor. This constructor takes a socket descriptor as a parameter. This means that you can pass a C socket descriptor to the constructor and the Socket class will represent the descriptor.

Next, let's go over the `sock.connects("127.0.0.1", 8888);` line. This function is a blocking call that connects the socket object to the server. The parameters of the function are a string representing the host and an integer representing the port on which the server is serving. Note: this function is a blocking call, so it will not terminate until it connects to the server. There is a variety of the function that is `Socket::connects();`. This function uses the data set from the `Socket(const char*, int);` constructor to connect to the server. This function returns an integer value on return. If the function fails, it should return -1; on success, 0. Upon failure, you can check `errno` for a more detailed cause of the error.

Finally, the `sock.closes();` method terminates the connection. It only returns void because the command rarely ever fails. If you really want, you can double check `errno` to check if an error has occured.

2.6 Accepting Connections

Next, let's talk about the `tcp::Server` class. This class is a simple blocking server that accepts `tcp::Socket` objects. Here's a basic example:

```
test.cc

   #include <iostream>
   #include <socketpp/tcp/server.h>
   #include <socketpp/tcp/socket.h>

   int main()
   {

     tcp::Server serv(8888);
     serv.binds();
```

```
    tcp::Socket& sock = serv.accepts();

    sock.closes();
    serv.closes();

    return 0;
  }
```

Yet again, let's go through this line by line:

1. At the start of the program, there are the same include statements as before, except this time the `socketpp/tcp/server.h` file is included. This file contains the `tcp::Server` class. Also, something to note is that in this example, I did not add a `using` statement. This is to show variety of how to write these programs. Also, you can see that there is another `main` method.
2.

   ```
   tcp::Server serv(8888);
   serv.binds();

   tcp::Socket& sock = serv.accepts();
   ```

   Here is the critical code of the program. It starts with the constructor which makes a new `tcp::Server` object. Note that in the constructor's parameters, there is an integer value. This value is the port on which the server will server. Similarly to the constructor in `tcp::Socket`, the connection data for the server can either be given in the constructor or the `Server::binds()` method.

   Next, the line `serv.binds();` is a blocking call that binds the server to the port. It is now server, but not listening. This means that it will not actually accept connections yet. This call returns 0 on success and !0 on failure. Check errno for a more detailed error report.

   Now, we get into the the `Server::accepts()` method. This method is a blocking call that listens for connections. This means that until a connection is recieved, the function does not terminate. Please take special note to the way the `Socket` class is written. **The amperstand (&) is necessary after the Socket type**. Without it, you will get an error. This is because the way that `iostream` class are derived. The `iostream` copy constructor is deleted, so the returned object must be returned as a pointer. You can check that this function succeeded by testing the `Socket::isConnected()` function, which returns true or false based on if it is connected or not.

A final method worth noting is the `Server::acceptFd()` function. Instead of returning a Socket object, the function returns the raw socket descriptor.

2.7 Sending and Recieving Data

Next, lets go over sending and recieving data. TCP connections are streams, which means that as one connection sends data, the other must recieve data. This is in contrast to UDP sockets, which simply broadcast their data. In short, this means that if one side of your program is sending data, the other mush be recieving.

The first step before sending or recieving data is making sure that your connections are established. See the previous sections for info on connecting to hosts. A good habit to make is checking that your sockets are properly connected before doing anything. Use the `Socket::isConnected()` meothod for doing this. Once this is done, you can proceed to send and recieve data.

Here is an example of sending and recieving data:

`server.cc`

```
#include <iostream>
#include <socketpp/tcp/socket.h>
#include <socketpp/tcp/server.h>

using namespace std;
using namespace tcp;

int main()
{

  // create and bind server
  Server serv;
  serv.binds(8888);

  // accept connection
  Socket& sock = serv.accepts();

  // make sure socket is connected
  if (!sock.isConnected())
     return -1;

  // read data
  char* buffer = new char[32];
  sock.get(buffer, 32);

  // print message
  cout << "Message from client: " << buffer << endl;

  // clean up
  delete buffer;
  sock.closes();
  serv.closes();

  return 0;
}
```

`client.cc`

```
#include <iostream>
#include <socketpp/tcp/socket.h>

using namespace std;

int main()
{

  // create and bind server
  Socket* sock;

  try {

      sock = new tcp::Socket("127.0.0.1", 8888, 1);

  } catch (ctor_exe_t& e) {
    cout << e.what() << endl;
  }


  // make sure socket is connected
  if (!sock->isConnected())
     return -1;

  // send data
  sock << "Hello World" << endl;

  sock.closes();

  return 0;
}
```

The output should be: `output`

```
Message from client: Hello World
```

Lets go over what is going on in these programs.

2.8 Setting Options

2.9 Closing Connections

3 Extending

4 License

0. PREAMBLE

   The purpose of this License is to make a manual, textbook, or other functional and useful document *free* in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

   This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

   We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

   This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

   A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

   A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

   The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "publisher" means any person or entity that distributes copies of the Document to the public.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

   If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

   If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

   If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

   It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

   You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

   A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.

B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
D. Preserve all the copyright notices of the Document.
E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
H. Include an unaltered copy of this License.
I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties—for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

   You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

   The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

   In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

   You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

   You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

   A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

   If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

   Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

   If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

   You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

   However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

   Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

   Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

    The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

    Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

    "Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

    "CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

    "Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

    An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

    The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

```
Copyright (C)  year  your name.
Permission is granted to copy, distribute and/or modify this document
under the terms of the GNU Free Documentation License, Version 1.3
or any later version published by the Free Software Foundation;
with no Invariant Sections, no Front-Cover Texts, and no Back-Cover
Texts.  A copy of the license is included in the section entitled ``GNU
Free Documentation License''.
```

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

```
with the Invariant Sections being list their titles, with
the Front-Cover Texts being list, and with the Back-Cover Texts
being list.
```

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.

Index

(Index is nonexistent)