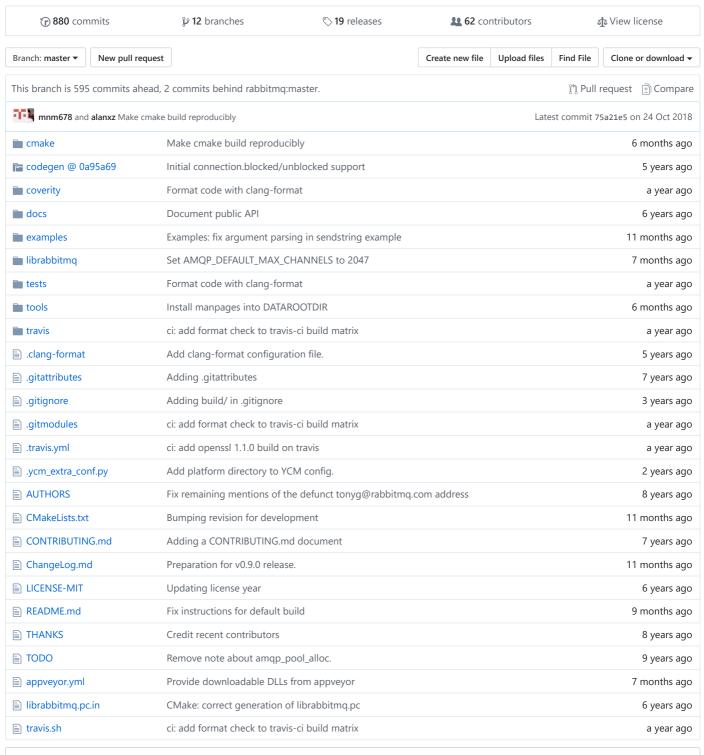
% alanxz / rabbitmq-c forked from rabbitmq/rabbitmq-c

RabbitMQ C client

#rabbitmq #rabbitmq-client



 $\boxplus \mathsf{README}.\mathsf{md}$

RabbitMQ C AMQP client library

build passing

coverage 43%

Introduction

This is a C-language AMQP client library for use with v2.0+ of the RabbitMQ broker.

• http://github.com/alanxz/rabbitmq-c

Announcements regarding the library are periodically made on the rabbitmq-c-users and cross-posted to rabbitmq-users.

- https://groups.google.com/forum/#!forum/rabbitmg-c-users
- https://groups.google.com/forum/#!forum/rabbitmq-users

Latest Stable Version

The latest stable release of rabbitmq-c can be found at:

• https://github.com/alanxz/rabbitmq-c/releases/latest

Documentation

API documentation for v0.8.0+ can viewed from:

http://alanxz.github.io/rabbitmq-c/docs/0.8.0/

Getting started

Building and installing

Prereqs:

- CMake v2.6 or better
- A C compiler (GCC 4.4+, clang, and MSVC are test. Other compilers may also work)
- Optionally OpenSSL v0.9.8+ to enable support for connecting to RabbitMQ over SSL/TLS
- Optionally POpt to build some handy command-line tools.
- Optionally XmlTo to build man pages for the handy command-line tools
- Optionally Doxygen to build developer API documentation.

After downloading and extracting the source from a tarball to a directory (see above), the commands to build rabbitmq-c on most systems are:

```
mkdir build && cd build
cmake ..
cmake --build . [--config Release]
```

The --config Release flag should be used in multi-configuration generators e.g., Visual Studio or XCode.

It is also possible to point the CMake GUI tool at the CMakeLists.txt in the root of the source tree and generate build projects or IDE workspace

Installing the library and optionally specifying a prefix can be done with:

```
cmake -DCMAKE_INSTALL_PREFIX=/usr/local ..
cmake --build . [--config Release] --target install
```

More information on CMake can be found on its FAQ (http://www.cmake.org/Wiki/CMake_FAQ)

Other interesting flags that can be passed to CMake:

- BUILD_EXAMPLES=ON/OFF toggles building the examples. ON by default.
- BUILD_SHARED_LIBS=ON/OFF toggles building rabbitmq-c as a shared library. ON by default.
- BUILD_STATIC_LIBS=ON/OFF toggles building rabbitmq-c as a static library. OFF by default.
- BUILD_TESTS=ON/OFF toggles building test code. ON by default.

- BUILD_TOOLS=ON/OFF toggles building the command line tools. By default this is ON if the build system can find the POpt header and library.
- BUILD_TOOLS_DOCS=ON/OFF toggles building the man pages for the command line tools. By default this is ON if BUILD_TOOLS is ON and the build system can find the XmlTo utility.
- ENABLE_SSL_SUPPORT=ON/OFF toggles building rabbitmq-c with SSL support. By default this is ON if the OpenSSL headers and library can be found.
- BUILD_API_DOCS=0N/OFF toggles building the Doxygen API documentation, by default this is OFF

Running the examples

Arrange for a RabbitMQ or other AMQP server to be running on localhost at TCP port number 5672.

In one terminal, run

```
./examples/amqp_listen localhost 5672 amq.direct test
```

In another terminal.

```
./examples/amqp_sendstring localhost 5672 amq.direct test "hello world"
```

You should see output similar to the following in the listener's terminal window:

```
Delivery 1, exchange amq.direct routingkey test
Content-type: text/plain
----
00000000: 68 65 6C 6C 6F 20 77 6F : 72 6C 64 hello world
0000000B:
```

Writing applications using librabbitmq

Please see the examples directory for short examples of the use of the librabbitmq library.

Threading

You cannot share a socket, an <code>amqp_connection_state_t</code>, or a channel between threads using <code>librabbitmq</code>. The <code>librabbitmq</code> library is built with event-driven, single-threaded applications in mind, and does not yet cater to any of the requirements of <code>pthread</code> ed applications.

Your applications instead should open an AMQP connection (and an associated socket, of course) per thread. If your program needs to access an AMQP connection or any of its channels from more than one thread, it is entirely responsible for designing and implementing an appropriate locking scheme. It will generally be much simpler to have a connection exclusive to each thread that needs AMQP service.