Installation

Jason L. Owens

June 15, 2016

Contents

1 Linux

2 Mac OS X

3 Windows

3.1 Building from source

Building Quickmsg from source on Windows requires significant development resources (e.g. decent hardware, a recent version of MS Visual Studio (no earlier than 2012)) and some additional software that must be downloaded, built and/or installed. The next sections will describe the steps to prepare your system for Quickmsg development, building dependencies, and finally building the Quickmsg libraries. As a developer, you may have your own versions of the suggested software already installed, or have a different way of doing things altogether. That's fine. This section is here to provide guidance when building Quickmsg so you can get up and running as quickly as possible.

3.1.1 Preparing your system

We assume you have created a development directory for your source code and libraries. In this document, we will refer to this directory as 'dev'. Since there are no good tools for managing build dependencies in Windows, we use the simplest method of collecting libraries into a single development directory and leverage relative paths in the build tools to help find the dependencies.

- Install 7z or your favorite zip program
- Install Git for Windows (`https://git-scm.com/download/win`)
- Install CMake (`https://cmake.org/download`)

  - Be sure to add cmake to the system path
- Download Boost 1.58 or later, unpack into `dev`
- Download TBB 4.4 or later, unpack into ‘dev‘ (`https://threadingbuildingblocks.org/download`)
  - Rename the TBB root directory to simply ‘tbb‘
- Install Oracle JDK 8
  - Set JAVA_HOME environment variable to the JDK directory
- Install Maven (`https://maven.apache.org/download.cgi`)
  - Add the ‘maven/bin‘ directory to the PATH environment variable
- Install Python 64-bit (`https://www.python.org/downloads/release/python-2711`)
  - Be sure to select the option to add python.exe to the PATH
  - If you want to custom-compile Python to get debug libraries, you need VS2010)
- Install SWIG 3.0 (for Windows) (`http://www.swig.org/download.html`)
  - Add the SWIG directory (containing swig.exe) to the PATH
- Install a version of MS Visual Studio (we’ve been using VS2013 Community Edition)

3.1.2 Getting and building the dependencies

- Download the Quickmsg repository into the `dev` directory
  - `git clone https://github.com/jlowenz/quickmsg.git`
- Build Boost using the included Quickmsg scripts
  - Launch a Visual Studio cmd prompt
  - Change to the boost root directory
  - run `bootstrap.bat`
  - run `..\quickmsg\msvc\compile_boost.bat`
  - set BOOST_ROOT environment variable to the root boost directory
- Download the ZeroMQ/Zyre dependencies
  - Launch a Git Bash session
  - run `quickmsg/msvc/get_zyre.sh` from the `dev` directory
- Build the Zyre dependencies
  - Launch a Visual Studio cmd prompt
  - Change to the `dev` directory
  - run `quickmsg/msvc/build_zyre.bat`

3.1.3 Building Quickmsg

- Configure Quickmsg
  - Create a `qmbuild` directory in the `dev` directory
  - run `cmake -G"Visual Studio 12 2013 Win64" -DCMAKE_BUILD_TYPE=Release ../quickmsg`
- Build Quickmsg
  - Change to `qmbuild` directory
  - run `msbuild /m /property:Configuration=Release ALL_BUILD.vcxproj`

NOTE: Although CMake produces a multi-configuration build setup, some values are only set in the build configuration step (thus, the CMAKE_BUILD_TYPE definition). This probably needs to be fixed somehow in the future.

3.2 Installing Binaries

...