LibVideo Beta

Documentation



Contents

[2 Introduction 3](#_Toc339979188)

[3 Installation and Setup 3](#_Toc339979189)

[3.1 Generating Visual Studios 2010 Solution (CMake) 3](#_Toc339979190)

[3.2 External Dependencies 4](#_Toc339979191)

[3.2.1 Live555 4](#_Toc339979192)

[3.2.2 FFMPEG 6](#_Toc339979193)

[3.2.3 Additional Dependencies 7](#_Toc339979194)

[4 Compiling in Ubuntu 7](#_Toc339979195)

# Introduction

Libvideo is an open-source software API written in C++ which allows users to generate [RTSP](http://en.wikipedia.org/wiki/Rtsp)-compliant video streams. To accomplish this, libvideo relies on two external frameworks: [live555](http://live555.com/) for handling RTSP and [ffmpeg](http://ffmpeg.org/) (or alternatively [Libav](http://libav.org/)) for handling video transcoding.

# Installation and Setup

## Generating Visual Studios 2010 Solution (CMake)

The current version of this software has been tested with Visual Studios 2010 as the standard software development platform, but may also work with earlier versions of Visual Studios. To build the libraries on your system, you will need [CMake](http://www.cmake.org/) (version 2.8.2 or greater) to generate the necessary .sln and .vsproj files to use with Visual Studio. The master CMakeLists.txt file is located in: (INSTALL\_PATH)/libraries/video/trunk/build/cmake). Follow the instructions below to create the necessary .sln and .vsproj files.

|  |  |
| --- | --- |
| 1. Open up the CMake-GUI application, and drag CMakeLists.txt into Cmake-GUI. 2. Modify the “Where to build the binaries” input to use a different directory, for instance: (INSTALL\_PATH)/libraries/video/trunk/build/cmake.msvc10 3. Then press the “Configure” button, and follow the guided dialog. 4. Press the “Generate” button. 5. The figure on the right illustrates the results of steps 1-4. Notice the “ Generating done” log entry in the bottom pane. |  |

Once steps 1-4 are complete, go to the earlier chosen “Where to build the binaries” folder to access the main solution for this software called “Video.sln.”

## External Dependencies

The easiest way to obtain all external dependencies for libvideo is simply to checked out the Active “3rdparty” folder from the subversion repository. If you are unable to access the “3rdparty” library, you must download or generate win32-compatible builds for live555, and ffmpeg (or Libav). This section details how this can be accomplished and gives a brief overview of the live555 and ffmpeg libraries.

### Live555

#### Overview

Libvideo requires Live555 to handle RTP, RTCP, and RTSP. At the top level, RTSP is used to handle control of streams (e.g. DESCRIBE, PLAY, PAUSE, RECORD, which are similar to VCR commands). RTP is the protocol handles the transmission of the compressed multimedia data, and RTCP is used to communicate stream quality control information.

Figure 1, Libvideo Dependencies

#### Setup

Verions of the prebuilt binary (win32) and source code for Live555can be found in the 3rdparty folder under (INSTALL\_PATH)/3rdparty/Live555. To build newer versions of Live555 for win32, please refer to the instructions below (based on this [link](http://tdistler.com/2011/05/14/live555-compiling-with-vs2008)).

##### Obtaining the Source

1. Download <http://www.live555.com/liveMedia/public/live555-latest.tar.gz>
2. Extract the contents obtained from (1) by using winrar, 7zip, or another archiver program.

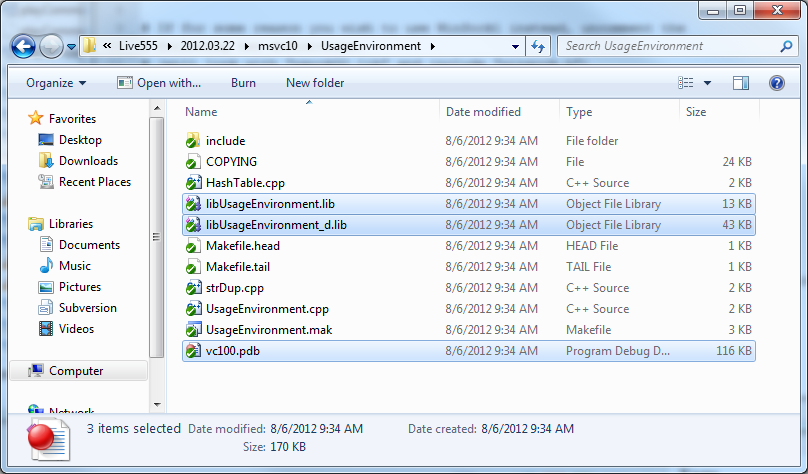
##### Building the Release Version

1. Go to the Live555 folder obtained from the previous section, 3.1.1.2.1.
2. Open the ‘win32config’ file and change the TOOLS32=... variable to your VS2008 install directory. This is set to “ TOOLS32=C:\Program Files (x86)\Microsoft Visual Studio 10.0\VC” if you are using Visual Studios 2010 installed in the standard location on Windows 7 64bit.
3. In ‘win32config’, modify the LINK\_OPTS\_0=... line from msvcirt.lib to msvcrt.lib. This fixes the link error:
   1. “LINK : fatal error LNK1181: cannot open input file 'msvcirt.lib'”
4. Open the Visual Studios command prompt. To open this go to:
   1. Start
   2. All Programs
   3. Microsoft Visual Studio 2010
   4. Visual Studio Tools
   5. Visual Studio Command Prompt (2010)
5. From the ‘live’ source directory, run genWindowsMakefiles
6. Now you’re ready to build. Simply run the following commands:
   1. cd liveMedia
   2. nmake /B -f liveMedia.mak
   3. cd ..\groupsock
   4. nmake /B -f groupsock.mak
   5. cd ..\UsageEnvironment
   6. nmake /B -f UsageEnvironment.mak
   7. cd ..\BasicUsageEnvironment
   8. nmake /B -f BasicUsageEnvironment.mak
7. By following step 1-4, you will have built the static release library for Live555

##### Building the Debug Version

If you have followed the previous section, 3.1.1.2.2, and have built the static release version, follow the instructions in 3.1.1.2.1 to create a new folder containing the uncompiled source. This must be done so that the debug binaries do not overwrite the previously built release binaries. After doing this, do the following:

1. Go to the Live555 folder obtained from the previous section, 3.1.1.2.1.
2. Open the ‘win32config’ file and change the TOOLS32=... variable to your VS2008 install directory. This is set to “ TOOLS32=C:\Program Files (x86)\Microsoft Visual Studio 10.0\VC” if you are using Visual Studios 2010 installed in the standard location on Windows 7 64bit.
3. **In ‘win32config’, change ‘NODEBUG=1’ to ‘NODEBUG=0’.**
4. In ‘win32config’, modify the LINK\_OPTS\_0=... line from msvcirt.lib to msvcrt.lib. This fixes the link error:
   1. “LINK : fatal error LNK1181: cannot open input file 'msvcirt.lib'”
5. Follow steps 4-6 for building the release version given in the previous section, 3.1.1.2.2.
6. By following steps 1-5 in this section, you have now obtained the debug static binaries for Live555. If you are only interested in building the debug version, you may skip the step 7.
7. Because the names of the debug and release binaries are named identically, the following must be done to merge both debug and release versions.
   1. Rename all binaries (\*.lib) in BasicUsageEnvironment, groupsock, UsageEnvironment, and liveMedia folder to end in “\*\_d.lib”.
      1. libBasicUsageEnvironment.lib → libBasicUsageEnvironment\_d.lib
      2. libgroupsock.lib → libgroupsock\_d.lib
      3. libliveMedia.lib → libliveMedia\_d.lib
      4. libUsageEnvironment.lib → libUsageEnvironment\_d.lib
   2. Copy all renamed debug libs (e.g. libgroupsock\_d.lib) to its respective folder in your release build. Nothing should be overwritten in this step.
   3. Copy all vc100.pdb files to its respective folder in your release build. Nothing should be overwritten in this step.
   4. Use your release build folder now as your merged build folder which now contains the Live555 source, debug and release binaries, and program debug databases. Use this folder’s handle as the Live555\_ROOT\_DIR for setting up libvideo with cmake, 3.1.
   5. The screenshot below depicts a view from the UsageEnvironment folder: you should have two .lib files and one .pdb file. The same should hold true in the groupsock, BasicUsageEnvironment, and liveMedia folders.



### FFMPEG

#### Overview

Libvideo requires the ffmpeg to handle compression and decompression of video data. The typical use in libvideo is to essentially use ffmpeg to encode or decode video data, which significantly reduces the required bandwidth that live555 uses to generate RTSP-compliant streams.

#### Setup

Ffmpeg can be replaced with libav at the user’s discretion. At the time of writing (4Q 2012), the latest builds of ffmpeg and libav have both been used successfully with libvideo without code change. You may choose to use one or the other, not both. However, it is highly recommended to try ffmpeg first as it has better support for win32.

FFMPEG is also part of the ACTIVE 3rdParty folder in the IST owned SVN if you have access to it.

##### Obtaining the Source and Win32 Binaries

Luckily, both the source and win32 binaries for ffmpeg (and libav) can be downloaded online. Check the table below for links and description.

|  |  |
| --- | --- |
| **Library** | **Link and Description** |
| ffmpeg | ([Link](http://ffmpeg.zeranoe.com/builds/))   1. Click the link above 2. Download the latest 32-bit “Shared” and “Dev” builds. The shared build contains dynamically linked libraries (.dll), while the dev build contains import libraries (.lib). 3. Extract both the “Shared” and “Dev” builds into the same folder, skip all conflicts. 4. The folder from the last step now contains debug and release dynamically linked libraries, import libraries, and header files. Use this folder’s handle as the Libav\_ROOT\_DIR for setting up libvideo with cmake, 3.1. |
| Libav | ([Link](http://win32.libav.org/win32/))   1. Click the link above. 2. Click on “Last modified” twice to sort by date, with newest files at the top. 3. Download the latest build, i.e., the first listed file under the time-sorted list. 4. Extract the contents of your download. 5. The folder from the last step now contains debug and release dynamically linked libraries, import libraries, and header files. Use this folder’s handle as the Libav\_ROOT\_DIR for setting up libvideo with cmake, 3.1. |

### Additional Dependencies

Libvideo requires its sister framework [CxUtils](http://active-ist.sourceforge.net/cxutils.php?menu=cxutils) to perform JPEG compression/decompression and its associated Image class. CxUtils can be found [here](http://active-ist.sourceforge.net/cxutils.php?menu=cxutils) with instructions on how to install and compile the binaries.

# Compiling in Ubuntu

### Downloading and installing Dependencies

From Zebulon new Ubuntu\_setup.txt file. To get general dependencies paste this line in terminal:

sudo apt-get install subversion build-essential libwxgtk2.8-dev libgtk2.0-dev libwxgtk2.8-dbg libdv4-dev libavc1394-dev libiec61883-dev libcv-dev libdc1394-22-dev libxtst-dev libhighgui-dev libv4l-dev libboost-all-dev libqt4-dev cmake-gui qtcreator liburg0-dev libsdl1.2-dev byobu yasm

Live555:

wget <http://www.live555.com/liveMedia/public/live555-latest.tar.gz>

Installation Directions taken from here:

<http://www.live555.com/liveMedia/#config-unix>

Unzip tar.gz file, go into folder in a terminal and type:

./genMakefiles <os-platform>

<os-platform> is replaced with the appropriate operating system. You can see what options are available in the live directory by listing the folders. If running Ubuntu, you will probably say linux, or linux-64bit. You can type uname -a to find your version of Ubuntu. If it returns anything like i686 or i386, then <os-platform> = linux, if you see x86\_64 then <os-platform> = linux-64bit

Once makefiles are generated, type make, and compilation should start. Generated files will be placed in the live directory.

LibAV:

This should be be installed on the system. If not, it can be installed with “sudo apt-get install libavcodec-dev”.

Cmake will find the correct directories, if not, you may have to find where your version is located and add it to the “FindLibav.cmake” file located in the build/cmake/libvideo folder.

Try using the ‘locate’ command in the terminal: “locate libavcodec”. Look for a libavcodec.a and libavcodec.so. Add those directory paths to the FindLibav.cmake file in the linux section. The line will look something like below, just add your path in the same format, under the existing path:

FIND\_PATH(Libav\_LIB\_DIR

NAMES

"libavcodec.a"

PATHS

"/usr/lib/x86\_64-linux-gnu"

)

If you are unable to compile at this point due to compilation errors, you may need to download and install a newer version of libav. Version 9 is the minimum compatible version with our library, it can be downloaded here:

<http://libav.org/releases/libav-9_beta2.tar.xz>

Download and extract the folder, configure, make and make install.

It’s possible you may need to include other libraries if linking errors occur during compilation.

Gsm --enable-libgsm

Schroedinger --enable-libschroedinger

Speex --enable-libspeex

Theoraenc --enable-libtheora

Vorbis --enable-libvorbis

Vpxenc --enable-libvpx

Matroskadec - unknown

Vaapi --enable-vaapi

First install them in Ubuntu

Sudo apt-get install libva-dev libschroedinger-dev libspeex-dev libtheora-dev libvorbis-dev libvpx-dev libgsm1-dev

Then reconfigure libav using

./configure --enable-libgsm --enable-libschroedinger --enable-libspeex --enable-libtheora --enable-libvorbis --enable-libvpx --enable-vaapi

Build and install the new files. Clear any old cmake files and re-run cmake. This may not solve the linking errors though. Certain example programs may not compile, but the majority should work.

### Compiling

Once dependencies are compiled and installed. You can first run cmake-gui to check that all dependencies are found. If not, go back to previous section.

Open qtcreator and open a project with the CmakeLists.txt file in the active/libraries/video/trunk/build/cmake folder. Run cmake, and the project should show up in qtcreator. Press the build button and everything will start compiling.