AVbin
=====

Home: http://avbin.github.com
Maintainer: Nathan Stocks <nathan.stocks@gmail.com>

End-User Installation

End-users should use the installers provided in the binary release. That is one
of the benefits of a binary distribution, after all...

Developing

AVbin is a C library following standard conventions.

Clone the git repository:

  git clone https://github.com/AVbin/AVbin.git avbin

See the header file ``include/avbin.h`` and the online API documentation at
http://avbin.github.com/AVbin/Docs.html

Building

If you are interested in testing, contributing code, or otherwise helping out,
we encourage you to join the Pyglet mailing list (AVbin was originally created
for the Pyglet project). See http://pyglet.org/contribute.html

AVbin is designed to be used with a particular version of Libav. This version
of Libav is included as a git submodule of the AVbin project under the ``libav``
directory. Note that this version of Libav may contain AVbin-specific patches,
though we try to get all patches accepted upstream.

Use the included `build.sh` script to compile AVbin. This takes care of
configuring Libav correctly, compiling it, then linking it with the AVbin
sources. The final AVbin libraries are placed in the ``dist`` directory.

To build on your platform, use the included ``build.sh`` script. See
``build.sh --help`` for usage information.

If you are planning on distributing the self-built binary to anyone other than
yourself, please create a "PRERELEASE" file in the root of the project directory
containing a version modifier such as "-my-version".
get_avbin_info()->version_string would then return "10-my-version" for a custom
build of AVbin 10, while get_avbin_info()->version will still return the integer
10.

Building for OS X

Tested on OS X Lion (10.7)

- Install Xcode (from Apple's App Store)
- Open Xcode, and go to ``Preferences > Downloads > Components`` and install the
  Command Line Tools package.
- Install yasm (You can install MacPorts and then run 'sudo port install yasm')
- Run ./build.sh [desired options and target]
- Obtain and install Auxiliary Tools for Xcode (from developer.apple.com >

Mac Dev Center > Developer Downloads)

Building for Linux

Tested on a default install of Ubuntu 12.04 LTS (both 32- and 64-bit installs).

- Install yasm: sudo apt-get install yasm
- Install libbz2-dev: sudo apt-get install libbz2-dev
- (If you want to make the binary installer)
  Install makeself: sudo apt-get install makeself
- Run ./build.sh [desired options and targets]

Building for Win64 or Win32 (crosscompiling on Linux)

Tested on Ubuntu 12.04 LTS 64-bit.

- Install MinGW-64: sudo apt-get install mingw-64
- (If you want to make the binary installer)
  Install nsis: sudo apt-get install nsis nsis-doc
- Cross-compile zlib and install it:

```
wget http://zlib.net/zlib-1.2.7.tar.gz
tar zxvf zlib-1.2.7.tar.gz
cd zlib-1.2.7

# For win32...
make PREFIX=i686-w64-mingw32- -f win32/Makefile.gcc
sudo make PREFIX=i686-w64-mingw32- \
  LIBRARY_PATH=/usr/i686-w64-mingw32/lib \
  INCLUDE_PATH=/usr/i686-w64-mingw32/include/ \
  BINARY_PATH=/dev/null \
  -f win32/Makefile.gcc install
make -f win32/Makefile.gcc clean

# For win64...
make PREFIX=x86_64-w64-mingw32- -f win32/Makefile.gcc
sudo make PREFIX=x86_64-w64-mingw32- \
  LIBRARY_PATH=/usr/x86_64-w64-mingw32/lib \
  INCLUDE_PATH=/usr/x86_64-w64-mingw32/include/ \
  BINARY_PATH=/dev/null \
  -f win32/Makefile.gcc install
```

- Cross-compile bzlib and install it

```
wget http://bzip.org/1.0.6/bzip2-1.0.6.tar.gz
tar zxvf bzip2-1.0.6.tar.gz
cd bzip2-1.0.6/

# For win32...
make CC=i686-w64-mingw32-gcc AR=i686-w64-mingw32-ar \
  RANLIB=i686-w64-mingw32-ranlib PREFIX=/usr/i686-w64-mingw32 libbz2.a
sudo cp libbz2.a /usr/i686-w64-mingw32/lib/
sudo cp bzlib.h /usr/i686-w64-mingw32/include/
make clean

# For win64...
make CC=x86_64-w64-mingw32-gcc AR=x86_64-w64-mingw32-ar \
  RANLIB=x86_64-w64-mingw32-ranlib PREFIX=/usr/x86_64-w64-mingw32 libbz2.a
sudo cp libbz2.a /usr/x86_64-w64-mingw32/lib/
sudo cp bzlib.h /usr/x86_64-w64-mingw32/include/
```

- (For creating dist zipfile) Install 7-Zip: sudo apt-get install p7zip-full

- Run ./build.sh [desired options and targets]

Caveats

AVbin is currently known to function on the following platforms

* Linux x86 (32- and 64-bit)

* Mac OS X 10.6 - 10.7 (32- and 64-bit)

* Windows XP (32-bit)

* Windows 7 (64-bit)

Creating a Distribution

- Using the same value for ``platform`` as the build you just did, run:
  ./dist.sh [platform]

Creating Documentation

- Install Doxygen from http://www.stack.nl/~dimitri/doxygen/

- Open/Edit ``Doxyfile`` to your liking and generate it with Doxygen. The GUI
  frontend seems to work just fine.

Installation and Usage

Run the binary installer, or place the resulting avbin.so, avbin.dylib or
avbin.dll from the ``dist`` directory into the appropriate system directory.

The AVbin dynamic library exports all of Libav's functions from libavcodec,
libavutil, libavformat, and libswscale. It also exports some higher-level
functions which have a fixed ABI (they will not change in incompatible ways
in future releases), documented in ``include/avbin.h``.

License

Due to the linkage between AVbin and Libav, AVbin must be licensed under the
LGPL or GPL. Currently all GPL features of the Libav configuration are
disabled, and AVbin is licensed under the LGPL.

You should see the accompanying COPYING.LESSER file for details. In summary,
you must note the usage of AVbin within the documentation of your application.
If you make changes to either library, you must either contribute said changes
upstream or include the sources of these changes within your distributed
application.

Contributions

Nathan Stocks   - Project maintainer
Alex Holkner   - Original author, original maintainer

Micah Richert - Contributed original macosx-x86-64 and win64 build scripts
Anatoly Techtonik - C++ integration
Jernej Virag - C code additions and updates, build script updates