Compiling using CMake

From OdaWiki

According to the Wikipedia page, CMake (http://en.wikipedia.org/wiki/CMake) is a unified, cross-platform, open-source build system that enables developers to build, test and package software by specifying build parameters in simple, portable text files. It works in a compiler-independent manner and the build process works in conjunction with native build environments, such as make, Apple's Xcode and Microsoft Visual Studio. It also has minimal dependencies, C++ only. CMake is open source software and is developed by Kitware.

Contents

Compatibility

- Windows
  - Visual C++ 2010 can compile the client, server and master without issue.
  - MinGW Makefiles can compile the client, server and master without issue.
  - Code::Blocks (MinGW) can compile the client, server and master without issues.
    - You **must** select a type of build (Debug/Release/etc.) when generating the project file, as the project does NOT have separate Debug and Release targets for every target. **This is by design and has no fix.** If you want to be able to switch between Debug and Release easily, keep two build directories.
- Mac OS X
  - Makefiles on Mac OS X 10.6 can compile the client, server, launchers and master.
- Linux
  - Makefiles can compile the client, server, launchers, and master without issue.
  - Code::Blocks uses Makefiles, so is equally as compatible.
- FreeBSD
  - Makefiles can compile the client, server and master without issue.

Windows

Installing CMake


cmake-gui running in Windows

The latest version of CMake can be downloaded from Kitware's website here (http://www.cmake.org/cmake/resources/software.html) .

Compiling Odamex

There are many options for compiling Odamex on Windows using CMake, and what you decide depends heavily on which IDE and compiler you have installed. If you don't have a compiler or IDE yet, you have a couple of choices:

- If you just want to compile Odamex with the least amount of hassle, **MinGW Makefiles** require the least amount of setup.
- If you want to use the same thing most other Odamex developers use, **Code::Blocks** is the recommended path.
- You can also use **Visual C++** if you prefer to use a Microsoft IDE and compiler.

There are many other generators available for CMake, however there are simply too many combinations and corner cases to cover in this wiki.

MinGW Makefiles

1. If you do not have MinGW already installed, follow the installation instructions for MinGW here (http://www.mingw.org/wiki/Getting_Started) .
2. Follow the installation instructions for any Required Libraries you might need.
3. Start up the CMake GUI tool.
4. In the input field labeled **Where is the source code:** pick out the folder where you checked out Odamex.
5. In the input field labeled **Where to build the binaries:** create a folder somewhere where you would like the Code::Blocks workspace to be created. If you're not sure where to put it, create a new folder called `build` in the folder where you checked out Odamex.
6. Click **Configure**.
   - If you get an error message at this point about a missing dll file, please re-read the installation instructions for MinGW here (http://www.mingw.org/wiki/Getting_Started) , particularly the part about adding `C:\MinGW\bin` to your `PATH` environment.
7. You should see a dialog box pop up. From the drop-down list, pick **MinGW Makefiles** and make sure **Use default native compilers** is selected, then click **Finish**.
8. After a few moments, you will see warnings about how SDL and SDL_mixer could not be found.
   - If you don't care about building the client, skip to the next step.
   - If you want to build the client, you need to set the **SDL_INCLUDE_DIR**, **SDL_LIBRARY_TEMP**, **SDL_MIXER_INCLUDE_DIR** and **SDL_MIXER_LIBRARY** cache variables. You need to check the "advanced" checkbox in order to see some of them. The INCLUDE_DIR entries points to the `include` directories of the libraries you just downloaded, **SDL_LIBRARY_TEMP** should point directly at the file `libSDL.dll.a` in the `lib` directory, and **SDL_MIXER_LIBRARY** should point directly at **SDL_mixer.lib**.
   - **If you have Odamex installed to C:\Program Files, read this!** For some reason, CMake will point to the libraries you have in C:\Program Files\Odamex instead of the correct path. Make sure that **SDLMIXER_LIBRARY** points to `SDL_mixer.lib`, and **SDL_LIBRARY** points to `libSDL.dll.a`. If they're not, change them so they are. All three of these files should be in the SDL library directories you just downloaded.
9. Click on the drop-down list next to **CMAKE_BUILD_TYPE** and select which type of build you would like to generate. Most likely, you will want to select Debug.
10. Click **Generate**.
11. Open up a Command Prompt and change to the `build` directory you created earlier.
12. Run the following command:
    - If you want to build everything: `mingw32-make`
    - If you want to build just the client: `mingw32-make odamex`
    - If you want to build just the server: `mingw32-make odasrv`
    - If you want to build just the master: `mingw32-make odamast`
    - If you want to clean up your build tree: `mingw32-make clean`

Code::Blocks

The configuration and generation steps are largely the same as above. However, when the CMake GUI asks you to select a generator, make sure and select **CodeBlocks - MinGW Makefiles**. And of course, instead of going to the command line and running `mingw32-make`, simply open the generated project with Code::Blocks.

Visual C++

1. Follow the installation instructions for any Required Libraries you might need.
2. Install a copy of Visual C++.
   - Visual C++ 2012 Express for Windows Desktop is free and more than capable of compiling Odamex. You can download it here (http://www.microsoft.com/visualstudio/eng/downloads#d-express-windows-desktop) .
   - If you are a college student, your university might be participating in MSDNAA. If so, you might be eligible for a copy of Visual Studio Professional for free. Check out this website (http://msdn.microsoft.com/en-us/academic) for details.
   - If you personally own any version of Visual Studio since 2005, Visual C++ is included on your Visual Studio DVD as Visual C++ has not been sold as a separate product since 2003.
   - Finally, if you feel like shelling out money for it, various editions of Visual Studio 2010 are available from Amazon (http://www.amazon.com/s/ref=nb_sb_ss_c_2_18?url=search-alias%3Dsoftware&field-keywords=visual+studio+2010&sprefix=visual+studio+2010) and Newegg (http://www.newegg.com/Product/ProductList.aspx?Submit=ENE&DEPA=0&Order=BESTMATCH&Description=Visual+Studio+2010&x=0&y=0) . Given all of the free alternatives this is not recommended, and if you do any serious development work on Windows you likely already have a copy.
3. Start up the CMake GUI tool.
4. In the input field labeled **Where is the source code:** pick out the folder where you checked out Odamex.
5. In the input field labeled **Where to build the binaries:** create a folder somewhere where you would like the Code::Blocks workspace to be created. If you're not sure where to put it, create a new folder called `build` in the folder where you checked out Odamex.
6. Click **Configure**.
7. You should see a dialog box pop up. From the drop-down list, pick:
   - **Visual Studio 11** if you have Visual C++ 2012.
   - **Visual Studio 10** if you have Visual C++ 2010.
   - **Visual Studio 9** if you have Visual C++ 2008.
   - **Visual Studio 8** if you have Visual C++ 2005.
8. Make sure **Use default native compilers** is selected, then click **Finish**.
9. After a few moments, you will see warnings about how SDL and SDL_mixer could not be found.
   - If you don't care about building the client, skip to the next step.
   - If you want to build the client, you need to set the **SDL_INCLUDE_DIR**, **SDL_LIBRARY_TEMP**, **SDL_MIXER_INCLUDE_DIR** and **SDL_MIXER_LIBRARY** cache variables. You need to check the "advanced" checkbox in order to see some of them. The INCLUDE_DIR entries points to the `include` directories of the libraries you just downloaded, **SDL_LIBRARY_TEMP** should point directly at the file `SDL.lib` in the `lib` directory, and **SDL_MIXER_LIBRARY** should point directly at **SDL_mixer.lib**.
   - **If you have Odamex installed to C:\Program Files, read this!** For some reason, CMake will point to the libraries you have in C:\Program Files\Odamex instead of the correct path. Make sure that **SDLMIXER_LIBRARY** points to `SDL_mixer.lib`, and **SDL_LIBRARY** points to `SDL.lib`. If they're not, change them so they are. All three of these files should be in the SDL library directories you just downloaded.
10. Click **Generate**.
11. Navigate to the `build` directory you created earlier and double click on the **Odamex.sln** solution file to open it in Visual C++.
12. Press F7 to build the entire project.

Running Odamex

The first time you run the client or server, you might run into some issues.

Client notes


A fully-populated client directory

The first time you run the client after building it, you will get an error message about a missing SDL.dll file. You need to copy:

- `SDL.dll` from the SDL Development Library's `lib` folder.
- All of the DLL files from the SDL_mixer Development Library's `lib` folder.
- `odamex.wad` from the base odamex SVN checkout folder.
- A DOOM IWAD from one of your installations of DOOM.

...into the folder where `odamex.exe` is. It is either located in the `client` subfolder of your build folder, or in one of the subfolders within `client`.

Server notes

The first time you build the server after building it, you will get an error message about not being able to find odamex.wad. You need to copy:

- `odamex.wad` from the base odamex SVN checkout folder.
- A DOOM IWAD from one of your installations of DOOM.

...into the folder where `odasrv.exe` is. It is either located in the `server` subfolder of your build folder, or in one of the subfolders within `server`.

Linux

Compiling Odamex using CMake has been tested on Debian Linux 6.0. The instructions for the other distributions have been inferred using available documentation. If you are having trouble with a specific configuration, please add a response to this bug (http://odamex.net/bugs/show_bug.cgi?id=284) .

Installing CMake

Depending on your Linux distribution, you may or may not have a copy of CMake in your software repository. Even if you do, the version that is available might not be up-to-date. The following distributions have a version of CMake 2.8, which is what the current build script requires.

- Debian 6.0: **CMake 2.8.2** `aptitude install cmake`
- Fedora 15: **CMake 2.8.4** `yum install cmake`
- openSUSE 11.4: **CMake 2.8.3** `zypper in cmake`
- Slackware 13.37 **CMake 2.8.4** `pkgtool`
- Ubuntu 10.04 LTS: **CMake 2.8.0** `apt-get install cmake`
- Ubuntu 11.04: **CMake 2.8.3** `apt-get install cmake`

The following distributions have an out-of-date version of CMake. You are welcome to bypass the version check and see if it still works. Assuming that there isn't too much breakage and workarounds needed to support it, "official" support for CMake 2.6 will be considered.

- CentOS 5.x: **CMake 2.6.4 (through EPEL (http://fedoraproject.org/wiki/EPEL))** `yum install cmake`
- Debian 5.0: **CMake 2.6.0** `aptitude install cmake`
- Red Hat Enterprise Linux 5.x: **CMake 2.6.4 (through EPEL (http://fedoraproject.org/wiki/EPEL))** `yum install cmake`
- Scientific Linux 5.x: **CMake 2.6.4 (through EPEL (http://fedoraproject.org/wiki/EPEL))** `yum install cmake`

If you do not have an up-to-date CMake, or would prefer to use the absolute latest version, both binary and source tarballs can be downloaded here (http://www.cmake.org/cmake/resources/software.html) .

Compiling Odamex

Once you have Odamex checked out from SVN, change to the directory where you checked it out. From there, the process is relatively simple:

Finished compiling!

```
mkdir build && cd build && cmake ..
```

You might see warnings about not being able to find SDL or SDL_mixer. If you are not interested in compiling the client, ignore the warnings. Otherwise, please see the Required Libraries page for instructions on how to install SDL and SDL_mixer and try again.

Once `cmake` finishes its job, run the following command:

- If you want to build everything: `make`
- If you want to build just the client: `make odamex`
- If you want to build just the server: `make odasrv`
- If you want to build just the master: `make odamast`
- If you want to clean up your build tree: `make clean`

Build Types

CMake gives you a choice of four build types. The default build type is Debug, but there are four choices:

- Debug: Debug information, -O1 optimization.
- Release: No debug information, -O3 optimization.
- RelWithDebInfo: Debug information, -O3 optimization. Useful for finding optimization bugs that only show up in Release.
- MinSizeRel: HOLY COW I'M TOTALLY GOING SO FAST OH F*** (http://funroll-loops.info/) .

To specify a build type, you need to pass it with your cmake command like so:

```
cmake .. -DCMAKE_BUILD_TYPE=Release
```

Using Clang + LLVM

If you prefer to use clang instead of gcc, you can:

```
export CC=/usr/bin/clang
export CXX=/usr/bin/clang++
cmake -D CMAKE_TOOLCHAIN_PREFIX=llvm- ..
```

Note that CMake bakes the `CC` and `CXX` variables into the generated makefile, so you do not have to export them again if you run the makefile in a fresh shell environment.

Alternate SDL installations

If you are testing Odamex against multiple SDL versions, you can do so like this (assuming you compiled it with --prefix=/opt/SDL-1.2.13)

```
cmake .. -DSDLDIR=/opt/SDL-1.2.13
```

If you want to use a custom SDL mixer as well, you can --prefix it into the same directory as your custom SDL and CMake will pick up on it automatically via SDLDIR. Otherwise, you can also manually specify SDL_mixer like so:

```
cmake .. -DSDLMIXERDIR=/opt/SDL_mixer-1.10
```

Obviously you can mix and match the two params as you please (stock SDL and custom SDL_mixer, stock SDL and custom SDL_mixer, etc).

GUI Tool

If you want a tool similar to cmake-gui on Windows, there is an ncurses tool that comes with CMake called `ccmake`. The command-line syntax for using it is the same as `cmake`, but it gives you a nice little graphical interface to double-check the cache file with.

Mac OS X

The latest version of CMake can be downloaded from Kitware's website here (http://www.cmake.org/cmake/resources/software.html) .

You can also install the latest version using your package manager of choice:

- Homebrew: `brew install cmake`

Compiling Odamex

See Compiling Odamex on Linux (http://odamex.net/w/index.php?title=Compiling_using_CMake&action=submit#Compiling_Odamex_2)

Xcode

CMake supports generating Xcode project files.

```
cmake -G Xcode ..
```

OS X 10.4 Tiger

CMake can configure the build to use the OS X 10.4 Tiger SDK. This is useful for supporting as wide a range of OS X versions as possible. It is also required in order to build for ppc or to build a universal binary that includes ppc support.

```
cmake -DCMAKE_OSX_DEPLOYMENT_TARGET=10.4 \
  -DCMAKE_OSX_SYSROOT=/Developer/SDKs/MacOSX10.4u.sdk \
  -DCMAKE_CXX_COMPILER=g++-4.0 ..
```

These options can also be used to support any other installed SDK when something other than the system default is desired.

Universal Binaries

CMake also supports building universal binaries. For example, if you were interested in building a universal binary with ppc and i386 support:

```
cmake -DCMAKE_OSX_ARCHITECTURES="ppc;i386" ..
```

Valid values for CMAKE_OSX_ARCHITECTURES are `ppc`, `i386`, `ppc64` and `x86_64`.

FreeBSD

FreeBSD has CMake in its ports tree as a port and a package:

- Package: `pkg_add -r cmake`
- Port: `cd /usr/ports/devel/cmake && make install clean`

Compiling Odamex

See Compiling Odamex on Linux (http://odamex.net/w/index.php?title=Compiling_using_CMake&action=submit#Compiling_Odamex_2)