Compilation

In some cases it is more preferable to compile StepMania from its source code than use an installer or binary build.

StepMania utilizes the Git distributed version control system, and the cmake build system.

Compilation on Linux

Install dependencies

On current versions of Debian and Ubuntu (since at least 9.0 and 17.04 respectively), the following commands will install the dependencies needed to compile StepMania 5.1.

```
sudo apt-get install build-essential
sudo apt-get install git cmake mesa-common-dev libglu1-mesa-dev libglew1.5-dev libxtst-dev libxrandr-d
```

Here is a more elaborated explanation on what you install:

We will update this list when we change requirements.

First lets get the basic build-essentials

```
sudo apt-get install build-essential
```

The `sudo` means we run the command as root which is the highest level of permission on your system,

`apt-get` is the package manager of ubuntu, We use the `install` argument to tell it to install the packages we need,

Now lets get 2 packages we need on every system

```
sudo apt-get install git cmake
```

We use git to download the sourcecode from example github, this makes a clone of the files on github and put them on our harddrive.

Cmake is the configuration system for our game, it checks what system you have and whats available for our compiler to use.

Now we have both those, we can get the libaries we need for compiling the actual thing.

Here is an overview and explanation of all the libs we use.

- mesa-common-dev - These are the main graphic libs to make the game work on your pc.
- libglu1-mesa-dev - This is the GLU sub graphic lib, It has GLU which we use together with GLEW
- libglew1.5-dev - This is the GLEW main graphic lib, we use this for OpenGL.
- libxtst-dev - This is the X11 package, We use xorg/x11 as our graphical interface/GUI on linux.
- libxrandr-dev - This is the XRandr package, This package makes it so it works with your display settings.
- libpng-dev - PNG package, we use this to load PNG images in the game.
- libjpeg-dev - JPG/JPEG, same as above, but for JPG/JPEG instead.
- zlib1g-dev - zlib is a compressing lib, It makes sure we can compress stuff to make it smaller.
- libbz2-dev - Same as zlib, its for compressing, but it uses a higher quality compression.
- libogg-dev - libogg is what we use for our ogg support, most audio files these days made for SM are .ogg.
- libvorbis-dev - Vorbis is the same as above, Its for our ogg support, We mainly use this cuz its licences.
- libc6-dev - The standard C libraries we use for compiling, don't worry to much about this one.
- yasm or nasm - YASM or NASM are are assemblers, most people prefer YASM because of its licence and rewrite.
- libasound-dev - libasound-dev is the main ALSA sound development files, we need this for ALSA sound.
- libpulse-dev - Same as above but for pulseaudio, pulseaudio runs on ALSA and most systems use this.
- binutils-dev - An collection of binary utilities we use to compile the game.
- libudev-dev - An package that contains more dev files for compiling.
- libva-dev -Video Acceleration development files.

```

```

```
sudo apt-get install mesa-common-dev libglu1-mesa-dev libglew1.5-dev libxtst-dev libxrandr-dev libpng-
```

The rest is optimal for a reason, and I'll explain why.

- libgtk2.0-dev - We use this to show the loading window at the start, its old and broken.
- libmad0-dev - libmad is the MP3 support, its has licencing issues, don't use if you want to sell SM as a game.

```
sudo apt-get install libgtk2.0-dev libmad0-dev
```

Retrieving source code

To retrieve the StepMania 5.1 source code, run the following command.

```
git clone --single-branch -b 5_1-new --depth=1 https://github.com/stepmania/stepmania.git
```

Building StepMania

Run the following commands in sequence

```
cd stepmania
git submodule update --init
cd Build
cmake -G 'Unix Makefiles' -DCMAKE_BUILD_TYPE=Release ..make -jN
```

Change the **N** to a number less than or equal to double the number of cores in your CPU.

In case you want to test features and obtain more detailed debug information, replace "Release" in the cmake command with "Debug".

Compilation on macOS

Building StepMania

First, you'll need to install Homebrew package manager. Then open the terminal, and run the following commands.

```
brew install cmake yasmgit clone --single-branch -b 5_1-new --depth=1
https://github.com/stepmania/stepmania.git
cd stepmania
git submodule update --init
cd Buildcmake -G 'Xcode' -DCMAKE_BUILD_TYPE=Release .. && cmake ..
```

In case you want to test features and obtain more detailed debug information, replace "Release" in the cmake command with "Debug".

After compilation is complete, just double click the .xcodeproj file that is in the Build folder

If you have an error like ***Command /usr/bin/codesign failed with exit code 1*** then it means that the project was able to build the application, but the name of the app is not the exact one with the code sign process. To fix this, simply change the "Executable file" on the XCode project to: **`"Stepmania-[The CMake Build type you've chosen]"`**.

Compilation on Windows

Windows dependencies

To compile StepMania on Windows, you'll require a few things, including

- CMake 3.0 or newer

> be sure to install it to your PATH during installation so you can execute it on the command line

- Git - there are several options, such as Git for Windows, GitKraken, and the GitHub Desktop client. As long as you have the command line tools available, you'll be fine. If you use the GitHub Desktop application, it installs a PowerShell instance called "Git Shell" that provides enhancements for navigating Git repositories.
- Visual Studio 2017 or 2019 (these instructions presume that you will be using the free Community edition)
  - When installing Visual Studio, you will require at least the following components/workloads:

    - Desktop development with C++ components
    - Visual C++ Tools for CMake
    - Windows 10 SDK
    - Windows 8.1 SDK and UCRT SDK
- The DirectX SDK
    - If you encounter error code S1023 whilst installing the DirectX SDK, run the following commands in an Administrator command prompt, install the DirectX SDK, and then reinstall Visual C++ 2010 SP1 runtimes.
        - `MsiExec.exe /passive /X{F0C3E5D1-1ADE-321E-8167-68EF0DE699A5}`
        - `MsiExec.exe /passive /X{1D8E6291-B0D5-35EC-8441-6616F567A0F7}`

Building StepMania

When you're ready, once again retrieve the source code and run the following commands (assuming you are running Git from the command line. If not, follow the instructions for your software);

```
git clone --single-branch -b 5_1-new --depth=1 https://github.com/stepmania/stepmania.gitcd
stepmaniacd Build
```

Visual Studio 2017.

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

Visual Studio 2019 or newer.

```
cmake -A win32 ..
```

After running the commands, CMake will generate Visual Studio solution files, which can then be used to compile StepMania from within Visual Studio.

The directory structure containing the source folder MUST NOT contain spaces in ANY of the folder names.

Revision #33

Created 5 years ago by Lirodon

Updated 4 years ago by Lirodon