Detailed FFmpeg build guide for VS2013

The built was made without any modification in FFmpeg source code

1. Setup MSYS2

- Connect to http://msys2.github.io/
- Download package
  - **"x86_64"** for 64-bit Windows,
  - **"i686"** for 32-bit Windows
- Run downloaded package – it should bring up a Wizard.
  - Install into desired directory (Ex: *C:\msys*)
- After install:
  - Let wizard start shell window

    (or)
  - start shell window running **msys2_shell.bat**
- run the following commands to update msys packages (if it looks like package stops without finishing, close shell, restart shell and run command again)
  - ***pacman -Sy pacman***
  - ***pacman –Syu***
  - ***pacman –Su***
- running these commands for me added **mingw32.exe** and **mingw64.exe** in the *C:\msys* directory

2. Setup Yasm

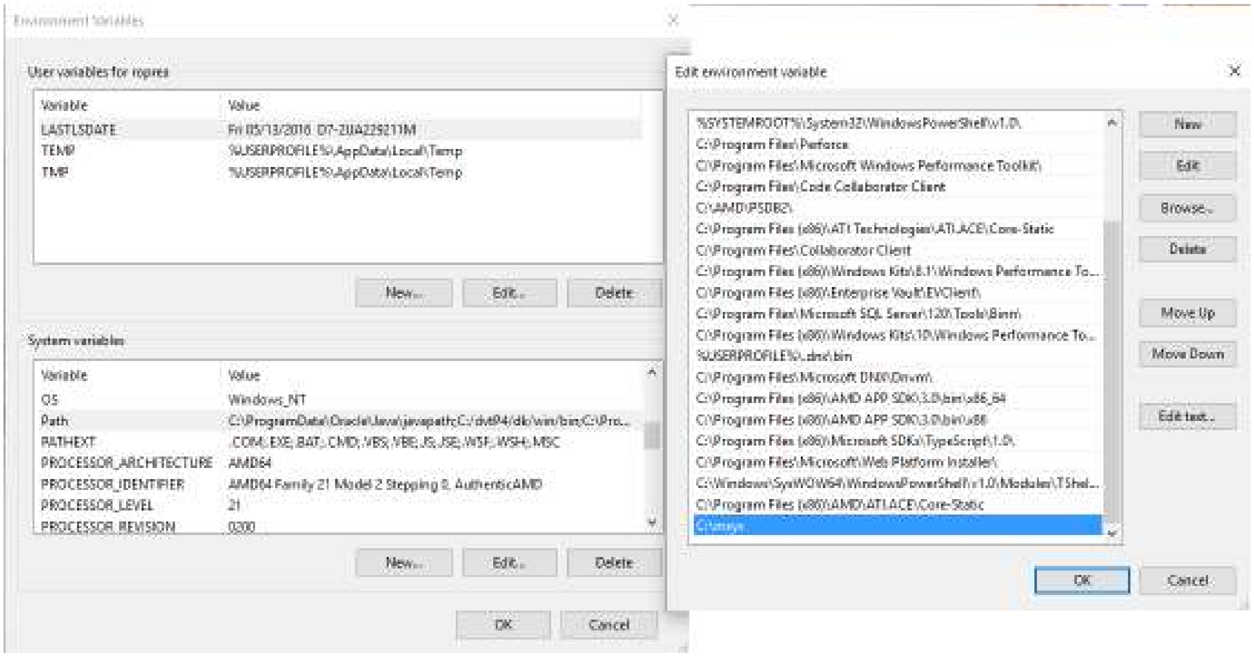
- Connect to http://yasm.tortall.net/Download.html
- Download package
  - **Win64.exe** (yasm-1.3.0-win64.exe) for 64-bit Windows
- Copy **yasm-1.3.0-win64.exe** to *C:\msys/***yasm.exe**
- During debug compilation it complained it can't find yasm (which is strange as release had no problems), so I also added **yasm.exe** to *C:\msys/usr/bin/***yasm.exe**

3. Rename Link.exe

Rename *C:\msys/usr/bin/***link.exe** to something else (Ex: *C:\msys/usr/bin/***link-original.exe**) so it doesn't conflict with Visual Studio **link.exe**

4. System PATH

Check if the path where MSYS2 was installed (Ex: *C:\msys*), was added to the system path. If it's not in the path, make sure to add it.



Environment Variables

User variables for ropres

Variable	Value
LASTLSDATE	Fri 05/13/2016 D7-20A229211M
TEMP	%USERPROFILE%\AppData\Local\Temp
TMP	%USERPROFILE%\AppData\Local\Temp

System variables

Variable	Value
OS	Windows_NT
Path	C:\ProgramData\Oracle\Java\javapath;C:/dvtP4/dk/win/bin;C:\Pro...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC
PROCESSOR_ARCHITECTURE	AMD64
PROCESSOR_IDENTIFIER	AMD64 Family 21 Model 2 Stepping 0, AuthenticAMD
PROCESSOR_LEVEL	21
PROCESSOR_REVISION	0200

Edit environment variable

```
%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\
C:\Program Files\Perforce
C:\Program Files\Microsoft Windows Performance Toolkit\
C:\Program Files\Code Collaborator Client
C:\AMD\PSDB2\
C:\Program Files (x86)\ATI Technologies\ATI.ACE\Core-Static
C:\Program Files\Collaborator Client
C:\Program Files (x86)\Windows Kits\8.1\Windows Performance To...
C:\Program Files (x86)\Enterprise Vault\EVClient\
C:\Program Files\Microsoft SQL Server\120\Tools\Binn\
C:\Program Files (x86)\Windows Kits\10\Windows Performance To...
%USERPROFILE%\.dnx\bin
C:\Program Files\Microsoft DNX\Dnvm\
C:\Program Files (x86)\AMD APP SDK\3.0\bin\x86_64
C:\Program Files (x86)\AMD APP SDK\3.0\bin\x86
C:\Program Files (x86)\Microsoft SDKs\TypeScript\1.0\
C:\Program Files\Microsoft\Web Platform Installer\
C:\Windows\SysWOW64\WindowsPowerShell\v1.0\Modules\TShel...
C:\Program Files (x86)\AMD\ATI.ACE\Core-Static
C:\maya
```

5. Installing pkg-config

NOTE: this step can probably be skipped – while the script/makefile complains that pkg-config is either not found or not working, it doesn't seem to affect compilation as the build seems to succeed.

- I found a good article at (http://www.gaia-gis.it/spatialite-3.0.0-BETA/mingw_how_to.html) describing how to install pkg-config so I'm going to include that as it's pretty descriptive:

pkg-config is a well known package configuration manager; it's widely used by many *open source* packages.
Unhappily **pkg-config** doesn't come already installed once you've installed **MinGW** and **MSYS**.
And, to make things worse, installing **pkg-config** on Windows is quite difficult and not at all straightforward.

> First of all, you must download pkg-config.exe from **GTK+ for Windows.**
> Then you can simply *unzip* this downloaded zip-file, and then copy the **pkg-config.exe** executable into **/MinGW/bin**
>
> That's not enough: this executable depends on the **GLib DLL**.
> So you must download GLib DLL too, always from **GTK+ for Windows**.
> Once again, you have to *unzip* this downloaded zip-file, and then copy **liglib-2.0-0.dll** into **/MinGW/bin**
>
> You have not yet finished: **pkg-config** still has an unresolved DLL dependency.
> But this time simply performing a trivial copy will suffit. So you must now open an MSYS shell:
>
> **cd C:/MinGW/bin**
> **cp libintl-8.dll intl.dll**
>
> And this time that's really all: now you have **pkg-confif** properly installed and ready to work.

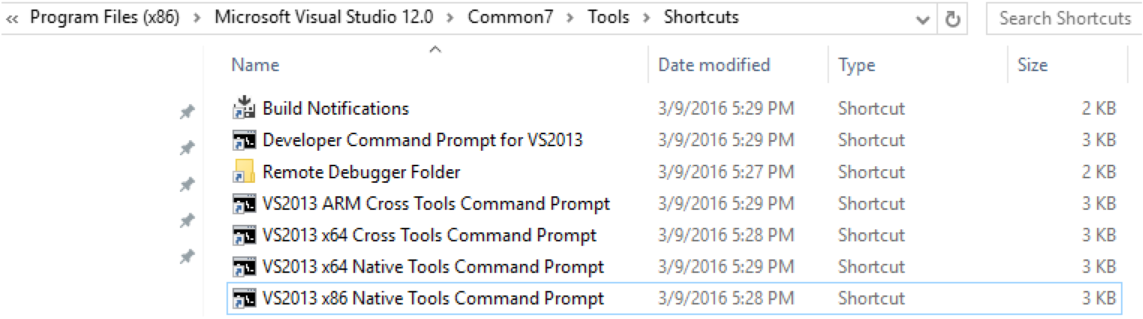
- The only thing with the above mentioned description is that the libintl-8.dll was nowhere to be found, however another file was there **msys-intl-8.dll**
- If the file (**msys-intl-8.dll**) cannot be found, the following package is available (under **\var\cache\pacman\pkg**): **libintl-0.19.7-3-x86_64.pkg.tar.xz**, which contains it
- Make a copy of the file to **intl.dll** (also leaving the original in place), and **pkg-config.exe** should be ready to go

6. Edit msys2_shell.bat

The current file looks like this:

```
:
@echo off

rem To activate windows native symlinks uncomment next line
rem set MSYS=winsymlinks:nativestrict

rem Set debugging program for errors
rem set MSYS=error_start:%WD%../../mingw32/bin/qtcreator.exe^|-debug^|^<process-id^>

rem To export full current PATH from environment into MSYS2 uncomment next line
rem set MSYS2_PATH_TYPE=inherit

call "%~dp0start_shell.cmd" -msys %*
:EOF
```

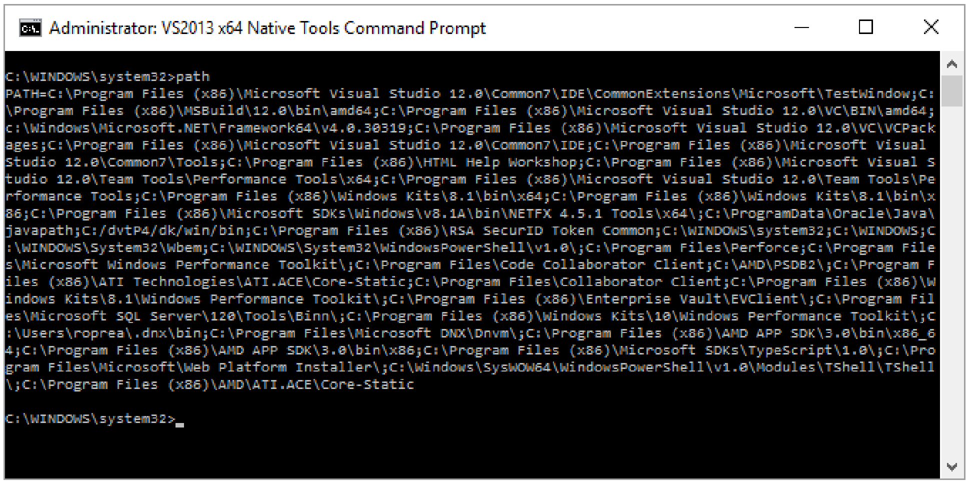
- Take out the **rem** from the *rem set MSYS2_PATH_TYPE=inherit* to inherit the path from the visual studio command line.
- Start <Visual Studio x64 Native Tools Command Prompt> (it's a good idea to run it as Administrator)
  - NOTE: To compile the 32 version of FFmpeg, make sure to start the 32 bit version of the tools < VS2013 x86 Native Tools Command Prompt> otherwise it will produce the 64 bit libraries even through the build script would suggest it's running 32 bit. By running the < VS2013 x86 Native Tools Command Prompt>, it will point to the 32 bit version of cl.exe while <Visual Studio x64 Native Tools Command Prompt> will point to the 64 bit version of cl.exe



« Program Files (x86) › Microsoft Visual Studio 12.0 › Common7 › Tools › Shortcuts

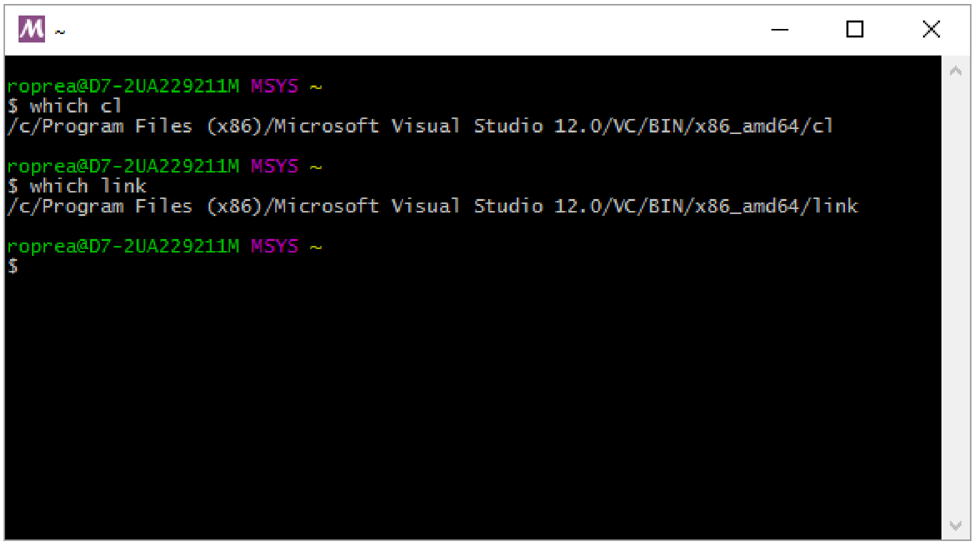
Name	Date modified	Type	Size
Build Notifications	3/9/2016 5:29 PM	Shortcut	2 KB
Developer Command Prompt for VS2013	3/9/2016 5:29 PM	Shortcut	3 KB
Remote Debugger Folder	3/9/2016 5:27 PM	Shortcut	2 KB
VS2013 ARM Cross Tools Command Prompt	3/9/2016 5:29 PM	Shortcut	3 KB
VS2013 x64 Cross Tools Command Prompt	3/9/2016 5:28 PM	Shortcut	3 KB
VS2013 x64 Native Tools Command Prompt	3/9/2016 5:29 PM	Shortcut	3 KB
VS2013 x86 Native Tools Command Prompt	3/9/2016 5:28 PM	Shortcut	3 KB

- If you type **path** in this command window you should see a nice long path to the Visual Studio paths (the path where MSYS2 was installed should also be there):



```
C:\WINDOWS\system32>path
PATH=C:\Program Files (x86)\Microsoft Visual Studio 12.0\Common7\IDE\CommonExtensions\Microsoft\TestWindow;C:
\Program Files (x86)\MSBuild\12.0\bin\amd64;C:\Program Files (x86)\Microsoft Visual Studio 12.0\VC\BIN\amd64;
c:\Windows\Microsoft.NET\Framework64\v4.0.30319;C:\Program Files (x86)\Microsoft Visual Studio 12.0\VC\VCPack
ages;C:\Program Files (x86)\Microsoft Visual Studio 12.0\Common7\IDE;C:\Program Files (x86)\Microsoft Visual
Studio 12.0\Common7\Tools;C:\Program Files (x86)\HTML Help Workshop;C:\Program Files (x86)\Microsoft Visual S
tudio 12.0\Team Tools\Performance Tools\x64;C:\Program Files (x86)\Microsoft Visual Studio 12.0\Team Tools\Pe
rformance Tools;C:\Program Files (x86)\Windows Kits\8.1\bin\x64;C:\Program Files (x86)\Windows Kits\8.1\bin\x
86;C:\Program Files (x86)\Microsoft SDKs\Windows\v8.1A\bin\NETFX 4.5.1 Tools\x64\;C:\ProgramData\Oracle\Java\
javapath;C:/dvtP4/dk/win/bin;C:\Program Files (x86)\RSA SecurID Token Common;C:\WINDOWS\system32;C:\WINDOWS;C
:\WINDOWS\System32\Wbem;C:\WINDOWS\System32\WindowsPowerShell\v1.0\;C:\Program Files\Perforce;C:\Program File
s\Microsoft Windows Performance Toolkit\;C:\Program Files\Code Collaborator Client;C:\AMD\PSDB2\;C:\Program F
iles (x86)\ATI Technologies\ATI.ACE\Core-Static;C:\Program Files\Collaborator Client;C:\Program Files (x86)\W
indows Kits\8.1\Windows Performance Toolkit\;C:\Program Files (x86)\Enterprise Vault\EVClient\;C:\Program Fil
es\Microsoft SQL Server\120\Tools\Binn\;C:\Program Files (x86)\Windows Kits\10\Windows Performance Toolkit\;C
:\Users\roprea\.dnx\bin;C:\Program Files\Microsoft DNX\Dnvm\;C:\Program Files (x86)\AMD APP SDK\3.0\bin\x86_6
4;C:\Program Files (x86)\AMD APP SDK\3.0\bin\x86;C:\Program Files (x86)\Microsoft SDKs\TypeScript\1.0\;C:\Pro
gram Files\Microsoft\Web Platform Installer\;C:\Windows\SysWOW64\WindowsPowerShell\v1.0\Modules\TShell\TShell
\;C:\Program Files (x86)\AMD\ATI.ACE\Core-Static

C:\WINDOWS\system32>
```

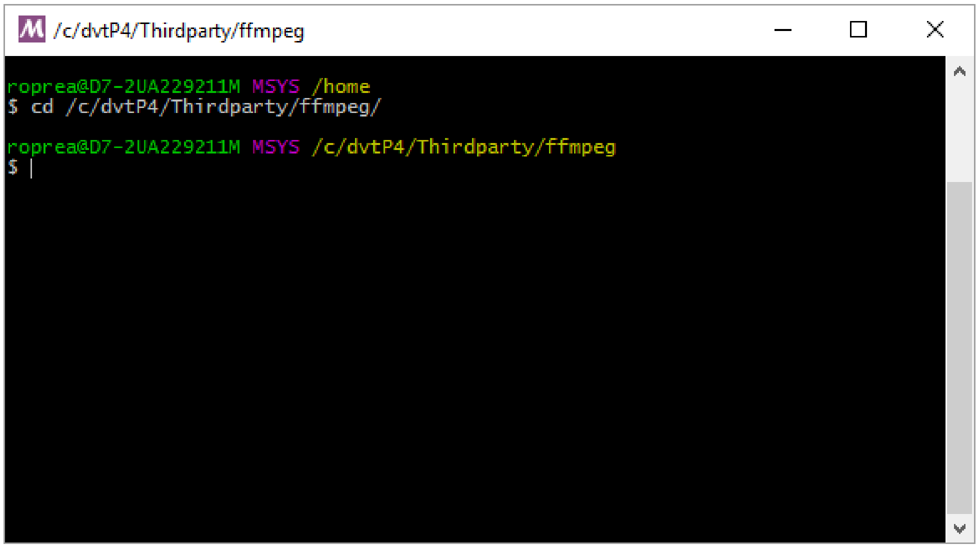
- From the Visual Studio Command Prompt, run the modified **msys2_shell.bat** file (Ex: *C:\msys/***msys2_shell.bat**).
- Type **which cl/link** in the shell window for msys to see which compiler and linker it will choose, just so we know we've done the proper steps so far. In the end it should look something similar to this:



```
roprea@D7-2UA229211M MSYS ~
$ which cl
/c/Program Files (x86)/Microsoft Visual Studio 12.0/VC/BIN/x86_amd64/cl

roprea@D7-2UA229211M MSYS ~
$ which link
/c/Program Files (x86)/Microsoft Visual Studio 12.0/VC/BIN/x86_amd64/link

roprea@D7-2UA229211M MSYS ~
$
```

7. Directory name change

At the msys command prompt, change directory to the path where ffmpeg source is located (Ex: *C:\dvtP4/Thirdparty/ffmpeg*) cd **/c/dvtP4/Thirdparty/ffmpeg**.

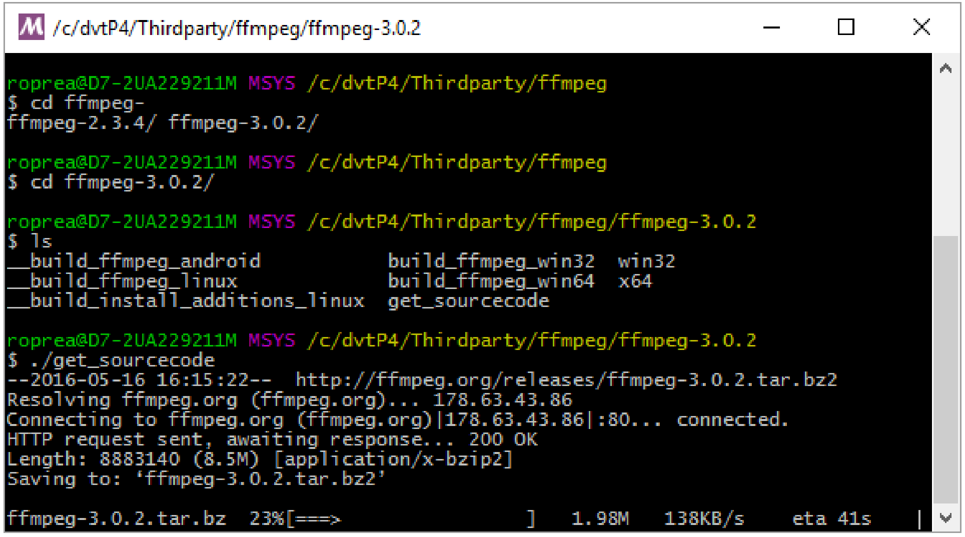


```
roprea@D7-2UA229211M MSYS /home
$ cd /c/dvtP4/Thirdparty/ffmpeg/

roprea@D7-2UA229211M MSYS /c/dvtP4/Thirdparty/ffmpeg
$ |
```

8. Getting source code

There is a script for getting the source code aptly named **get_sourcecode**.

- you will need to have two programs **wget** and **tar**, and you can check if they're availabe using **which wget/tar**.
- If **wget** and **tar** are not available, you will need to get them:
  - ***pacman –S wget***
  - ***pacman –S tar***
- run **./get_sourcecode** to get the version of FFmpeg (in this case 3.0.2) – it should show something along the lines:



```
roprea@D7-2UA229211M MSYS /c/dvtP4/Thirdparty/ffmpeg
$ cd ffmpeg-
ffmpeg-2.3.4/ ffmpeg-3.0.2/

roprea@D7-2UA229211M MSYS /c/dvtP4/Thirdparty/ffmpeg
$ cd ffmpeg-3.0.2/

roprea@D7-2UA229211M MSYS /c/dvtP4/Thirdparty/ffmpeg/ffmpeg-3.0.2
$ ls
__build_ffmpeg_android           build_ffmpeg_win32  win32
__build_ffmpeg_linux             build_ffmpeg_win64  x64
__build_install_additions_linux  get_sourcecode

roprea@D7-2UA229211M MSYS /c/dvtP4/Thirdparty/ffmpeg/ffmpeg-3.0.2
$ ./get_sourcecode
--2016-05-16 16:15:22--  http://ffmpeg.org/releases/ffmpeg-3.0.2.tar.bz2
Resolving ffmpeg.org (ffmpeg.org)... 178.63.43.86
Connecting to ffmpeg.org (ffmpeg.org)|178.63.43.86|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 8883140 (8.5M) [application/x-bzip2]
Saving to: ‘ffmpeg-3.0.2.tar.bz2’

ffmpeg-3.0.2.tar.bz  23%[===>                ]  1.98M   138KB/s    eta 41s    |
```

- the source code should've been downloaded and extraced under **./src/ffmpeg-3.0.2**
- this script will be run as part of "build" scripts (like **build_ffmpeg_win64**) so there's no need to run it manually from now on.

NOTE: there is a **tar** file containing the source code – this can be unpacked using **WinRar**, to eliminate the need to get the source code repeatedly. The code should get unpacked under **./src/ffmpeg-3.0.2** which is where the build script is looking for files. The final path, based on the screen above would be: *C:/dvtP4/Thirdparty/ffmpeg/ffmpeg-3.0.2/**scr/ffmpeg-3.0.2***

9. Build

We’re ready to finally compile the library. There are a number of scripts to run windows/linux builds.

- Type the corresponding script you would like to build (Ex: **./build_ffmpeg_win64**) and everything should be set in motion now.
- The build_ffmpeg_xxxxxx scripts contain more or less three lines of information:

```
#!/bin/bash
./get_sourcecode
./scripts/ffmpeg-build-win release 64
#./scripts/ffmpeg-build-win debug 64
```

- If you’ve already unpacked the source code you can comment out the **./get_sourcecode** command
- you can also comment out either the **debug/release** builds in case one is not needed.
  NOTE: the **debug** build is in fact **release** with some optimizations turned off