Lecture 03

- Problem Set 1 will be out next Monday and due Friday 2/5.
- Problem Set 2 will be out Wednesday 2/3, and then the rest will be two-week cycle.

- Get your SVN password from a TA or me during the office hour.

Setting up the environment

- This course uses a sub-version server for distributing program frameworks, sample codes, and submitting assignments.

- Let’s make an alias for the server address, so that your computer will look at the new server location by changing only one thing.

- In case of the server address change, you can just change the alias and everything will work the same, as if nothing has changed.

Setting up the environment

Windows

1. Open start menu and type notepad.
2. Right-click on the notepad icon, and select “Run as Administrator”, then click “Yes” on the warning.
3. File -> Open -> Move down to:
   C:\windows\system32\drivers\etc
4. Type *lmhosts* in the file-name box, and open.
5. If you see an error message “File not found”,
   1. Open *lmhosts.sam* instead, and choose File->SaveAs
   2. In the file-name box, type
      "lmhosts"
      <u>including the double-quotes</u>. If you don’t include double quote, notepad will kindly add .txt extension for you, and it won’t work.

Windows (Continued)

6. Type the following at the bottom of the file.
   128.2.57.155 ramennoodle
7. File -> Save
8. Close the notepad.
9. Open Start menu, type “cmd”
10. Right-click on the cmd.exe icon, and select “Run as Administrator”, and then click “Yes”
11. Type:
    nbtstat -R
    and enter.
12. Close the command prompt.

(*) Instead of steps 9 through 12, you can just restart your computer.

Setting up the environment

Mac OSX

1. In Finder, Go -> Go to Folder
2. Type in:
   /private/etc/hosts
3. Locate the file called hosts, and copy to the desktop
4. Open TextEdit, and open hosts file on the desktop.
5. Add in the bottom:
   128.2.57.155 ramennoodle
6. Then save.
7. Drag & drop hosts file to the window of /private/etc.
8. Click on Authenticate, and click <u>Replace</u>
9. Restart is not supposed to be required, but to be safe, restart MacOSX.

Setting up the environment

MacOSX alternative: If you know how to use a text editor (like emacs) from the terminal

1. Type:

   sudo emacs /private/etc/hosts

2. Edit the hosts file as explained in the previous page.
3. Save the file by pressing:

   Ctrl-X, Ctrl-S

emacs

- Emacs is a very user unfriendly editor, but it's been there for long time, and there is a large user base. So, if you learn how to use it, it's going to last for another a few decades.
- It is very user unfriendly, but more friendly than other built-in text editors of Unix-based systems.
- Useful shortcuts:

Ctrl-X, Ctrl-F	Open a file
Ctrl-X, Ctrl-S	Save
Ctrl-X, Ctrl-W	Save as
Ctrl-Space	Mark set
Ctrl-W	Cut between the cursor and the mark
Ctrl-Y	Paste
Ctrl-K	Cut one line
	(Surprisingly there is no Copy)
Ctrl-F	Cursor forward
Ctrl-B	Cursor backward
Ctrl-P	Cursor previous line
Ctrl-N	Cursor next line

Test your alias

- Open a web browser, and type
  https://ramennoodle
- You may see the warning (because I cannot afford a Server Certificate!), but just proceed.
- Type in your AndrewID and the password given from the CA, you will see VisualSVN server.

Development Tools

- Version-Controlling System: SubVersion
- Cross-Platform Make: CMake

Version-Controlling System

- Keeps source files in the repository.
- Keeps track of changes. (Solution to: Oh no! My program was working yesterday!)
- Particularly useful when you are developing a program in a team.
- You can use it not just for a programming project, but for anything.
  - Research paper
  - Modeling
  - Simulation

Version-Controlling System

- A little bit of change of paradigm: The newest version code is (should be) kept in the repository, not in your disk drive.
- You can (should be able to) check out, or update, the working copy from the repository and work from any computer.
- If you mess up and lost track of what you did to the code, you can casually delete (I recommend to rename rather) the source code and re-check out and continue working.
- Your working copy in the local disk drive is good for a back up copy, in case of catastrophic server failure.

Popular Version-Controlling System

- CVS
  - Old version-controlling system. Only one person could work on the same part of the repository. (Locking/Unlocking)
- SubVersion (svn)
  - Well tested and gets the job done.
  - Very simple and easy to use.
- Git
  - More sophisticated.
  - Local-commit capability (Locally tracking changes before pushing it to the main repository.)

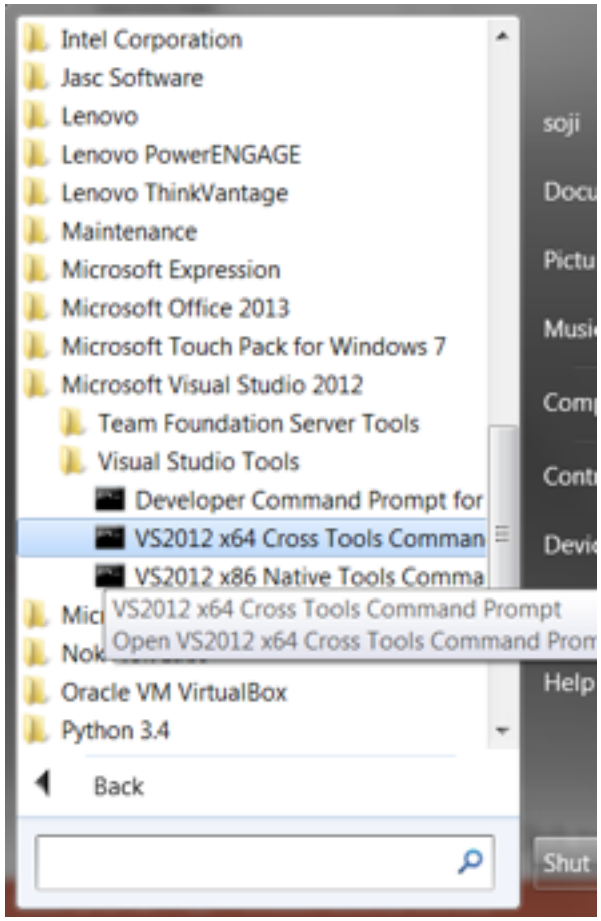
Setting up SubVersion

- Windows
  - https://tortoisesvn.net/
  - Make sure to install command-line tools and set system path. (We use a lot of command line!)

- Mac OSX
  - SVN comes with XCode Command-Line Tools.

Checking out source files

- Open terminal or command prompt.
  - In Windows, open Visual Studio x64 Command Prompt.
  - In MacOSX, find “Terminal.app” from the spotlight search.



Checking out source files (Windows)

VS2012 x64 Cross Tools Command Prompt - svn checkout https://ramennoodle/svn/public/public public

```
C:¥Program Files (x86)¥Microsoft Visual Studio 11.0¥VC>cd /D "%USERPROFILE%"

D:¥Users¥soji>mkdir eng_comp

D:¥Users¥soji>cd eng_comp

D:¥Users¥soji¥eng_comp>mkdir src

D:¥Users¥soji¥eng_comp>cd src

D:¥Users¥soji¥eng_comp¥src>svn checkout https://ramennoodle/svn/public/public public
Error validating server certificate for  https://ramennoodle:443 :
 - The certificate is not issued by a trusted authority. Use the
   fingerprint to validate the certificate manually!
 - The certificate hostname does not match.
Certificate information:
 - Hostname: NUTHATCH
 - Valid: from Sep 11 13:00:32 2015 GMT until Sep  8 13:00:32 2025 GMT
 - Issuer:
 - Fingerprint: E3:10:B4:1B:02:C0:D5:E8:D0:9B:7C:FE:05:CF:62:84:BF:C9:E3:06
(R)eject, accept (t)emporarily or accept (p)ermanently? p
Authentication realm: <https://ramennoodle:443> VisualSVN Server
Password for 'soji': _
```

Checking out source files (Windows)

```
VS2012 x64 Cross Tools Command Prompt
A    public¥iOS¥ysclass.xcodeproj
A    public¥iOS¥ysclass.xcodeproj¥project.pbxproj
A    public¥iOS¥ysclass.xcodeproj¥project.xcworkspace
A    public¥iOS¥ysclass.xcodeproj¥project.xcworkspace¥contents.xcworkspacedata
A    public¥iOS¥ysclass11.xcodeproj
A    public¥iOS¥ysclass11.xcodeproj¥project.pbxproj
A    public¥iOS¥ysfontrenderer.xcodeproj
A    public¥iOS¥ysfontrenderer.xcodeproj¥project.pbxproj
A    public¥iOS¥ysgl.xcodeproj
A    public¥iOS¥ysgl.xcodeproj¥project.pbxproj
A    public¥iOS¥ysglcpp.xcodeproj
A    public¥iOS¥ysglcpp.xcodeproj¥project.pbxproj
A    public¥iOS¥ysport.xcodeproj
A    public¥iOS¥ysport.xcodeproj¥project.pbxproj
A    public¥iOS¥yssystemfont.xcodeproj
A    public¥iOS¥yssystemfont.xcodeproj¥project.pbxproj
A    public¥iOS¥ignoreuserdata.py
Checked out revision 409.

D:¥Users¥soji¥eng_comp¥src>dir
 Volume in drive D is Work
 Volume Serial Number is F231-53C0

 Directory of D:¥Users¥soji¥eng_comp¥src

12/22/2015  11:46 AM    <DIR>          .
12/22/2015  11:46 AM    <DIR>          ..
12/22/2015  11:46 AM    <DIR>          public
               0 File(s)              0 bytes
               3 Dir(s)  92,282,064,896 bytes free

D:¥Users¥soji¥eng_comp¥src>_
```

Checking out source files (Mac OSX, Linux)

src — -bash — 80×49

```
SOJI-HUMMINGBIRD:~ soji$ cd ~
SOJI-HUMMINGBIRD:~ soji$ mkdir eng_comp
SOJI-HUMMINGBIRD:~ soji$ cd eng_comp
SOJI-HUMMINGBIRD:eng_comp soji$ mkdir src
SOJI-HUMMINGBIRD:eng_comp soji$ cd src
SOJI-HUMMINGBIRD:src soji$ svn checkout https://ramennoodle/svn/public/public public

Error validating server certificate for 'https://ramennoodle:443':
 - The certificate is not issued by a trusted authority. Use the
   fingerprint to validate the certificate manually!
 - The certificate hostname does not match.
Certificate information:
 - Hostname: NUTHATCH
 - Valid: from Fri, 11 Sep 2015 13:00:32 GMT until Mon, 08 Sep 2025 13:00:32 GMT
 - Issuer: NUTHATCH
 - Fingerprint: e3:10:b4:1b:02:c0:d5:e8:d0:9b:7c:fe:05:cf:62:84:bf:c9:e3:06
(R)eject, accept (t)emporarily or accept (p)ermanently? p
Authentication realm: <https://ramennoodle:443> VisualSVN Server
Password for 'soji':
A    public/src
A    public/src/test
A    public/src/test/interactive
A    public/src/test/interactive/ysglslcpp
A    public/src/test/interactive/ysglslcpp/main.cpp
```

```
A    public/iOS/ysglcpp.xcodeproj
A    public/iOS/ysglcpp.xcodeproj/project.pbxproj
A    public/iOS/ysport.xcodeproj
A    public/iOS/ysport.xcodeproj/project.pbxproj
A    public/iOS/yssystemfont.xcodeproj
A    public/iOS/yssystemfont.xcodeproj/project.pbxproj
A    public/iOS/ignoreuserdata.py
Checked out revision 410.
SOJI-HUMMINGBIRD:src soji$ ls
public
SOJI-HUMMINGBIRD:src soji$
```

CMake

- CMake – Cross-Platform Make
- You can write a program, but what about a build-environment?
- Too many build tools:
  - Make Good for Unix-based systems
  - Visual Studio Good for Windows (Trying to swallow others, too)
  - Xcode Good for MacOSX and iOS
  - Eclipse
  - Qt
- You cannot maintain a build environment for each platform after the number of source code gets hundreds.
- CMake is a solution.

CMake

- CMake interprets scripts called CMakeLists.txt, which is placed in each source directory, and generates project files for Make, Visual Studio, or XCode.
- Then you can build a program using the project file to build your program.

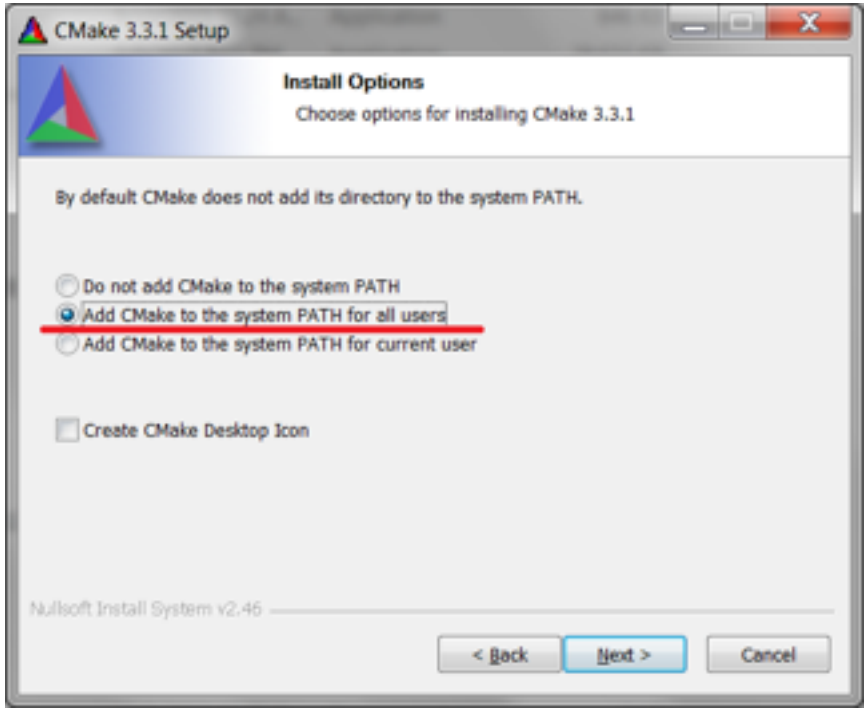
- Uses a concept of “Out-Of-Source” build.

Out-Of-Source Build

- <-> In-Source Build, which creates intermediate files and executable binaries in the source directory (or a sub-directory of the source directory).
- In-Source Build causes back-up nightmare.
  - Intermediate files can be huge!
  - We can re-create intermediate files from the source, we don't need to back them up.
  - Want to keep intermediate files and executable files separate from the source files.
- Out-Of-Source Build
  - Makes a separate build directory.

Installing CMake

- Download CMake from the following URL:
  https://cmake.org/
  and install.
- In Windows, make sure to choose “Add CMake to the system PATH for all users” during the installation. (If you forget, install it again.)



Installing CMake

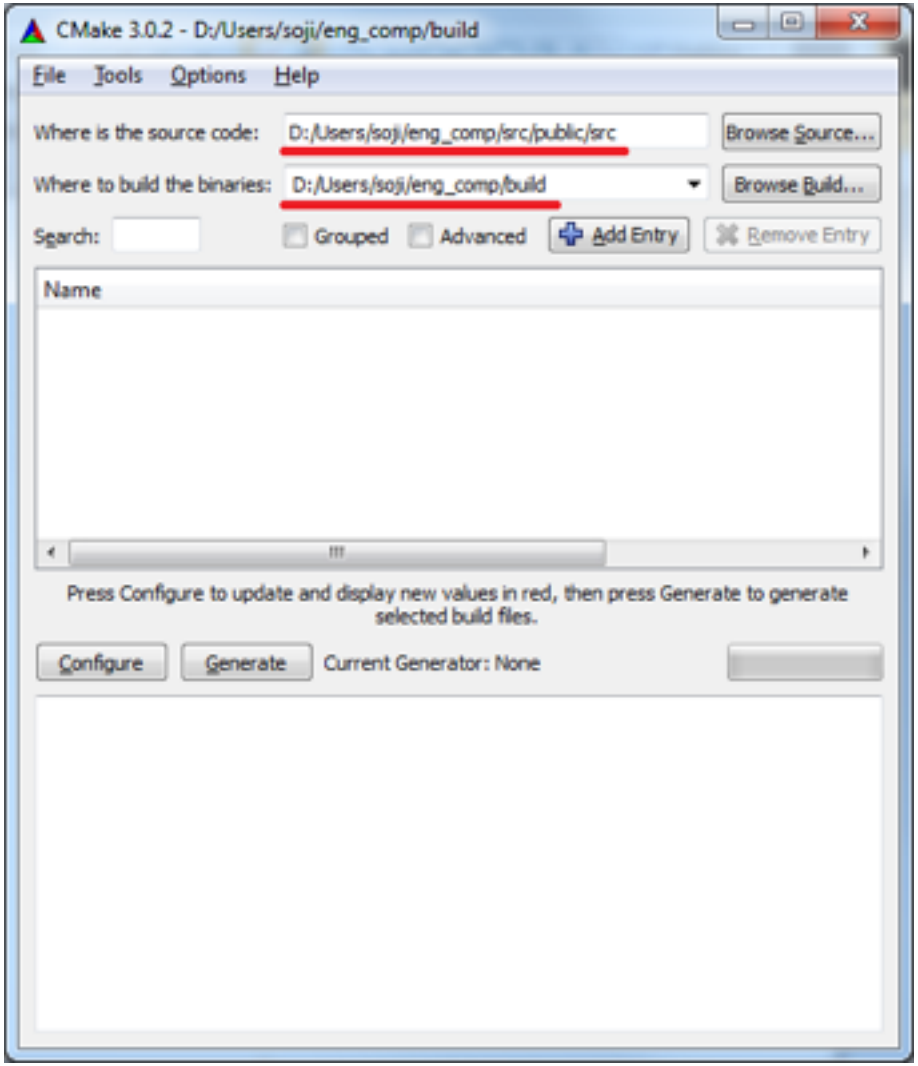
- In MacOSX, install CMake (drag & drop CMake App to Applications), and then
- Open Terminal, and type:

- Then follow the instructions.

Running CMake

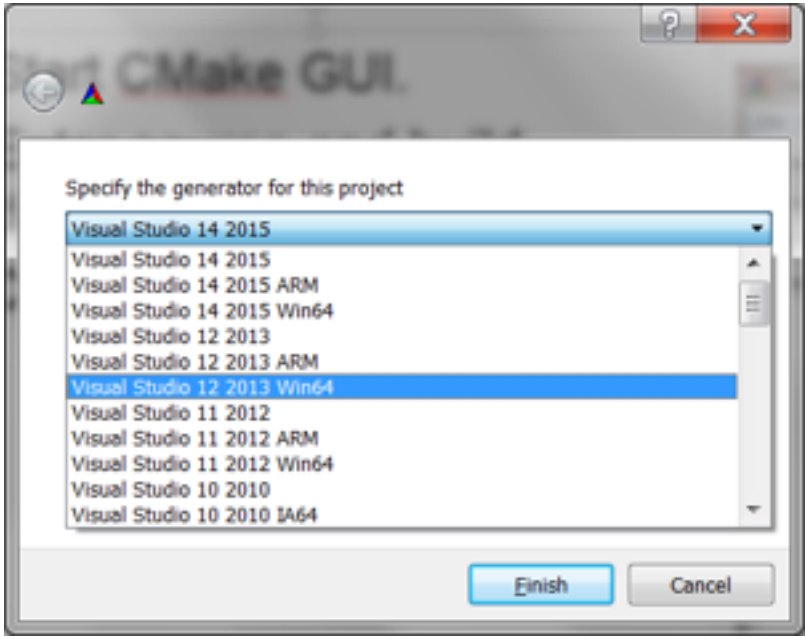
In Windows

- Start CMake GUI.
- Enter source and build directories.
- Click on “Generate”



Running CMake

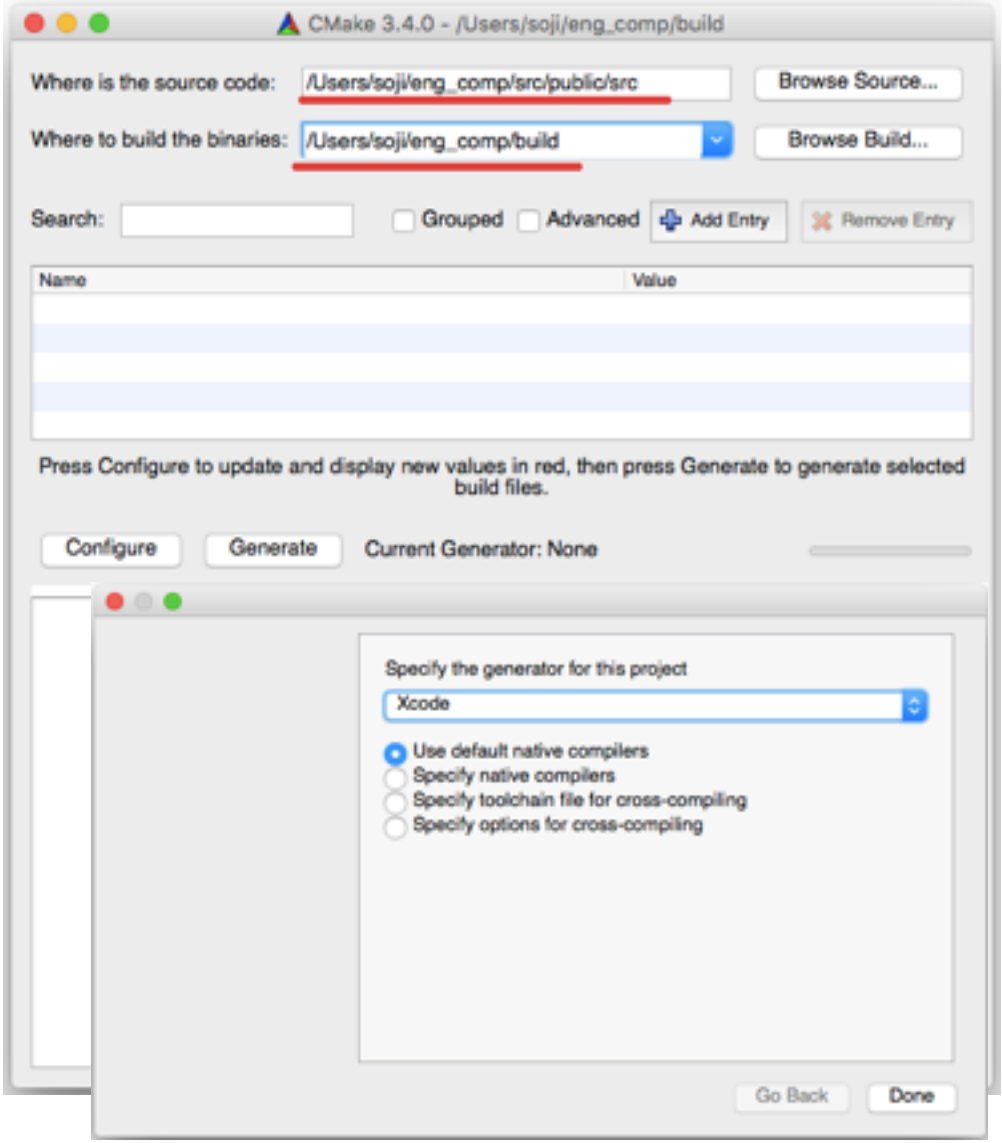
- Select Generator and Click on “Finish”
- Visual Studio Version 11 is Visual Studio 2012. Don’t get confused. (Subtract 2001 from the Visual Studio year to convert to the Version.)



Running CMake

In MacOSX

- Enter source and build locations, and click on Generate
- Select Xcode as the generator.

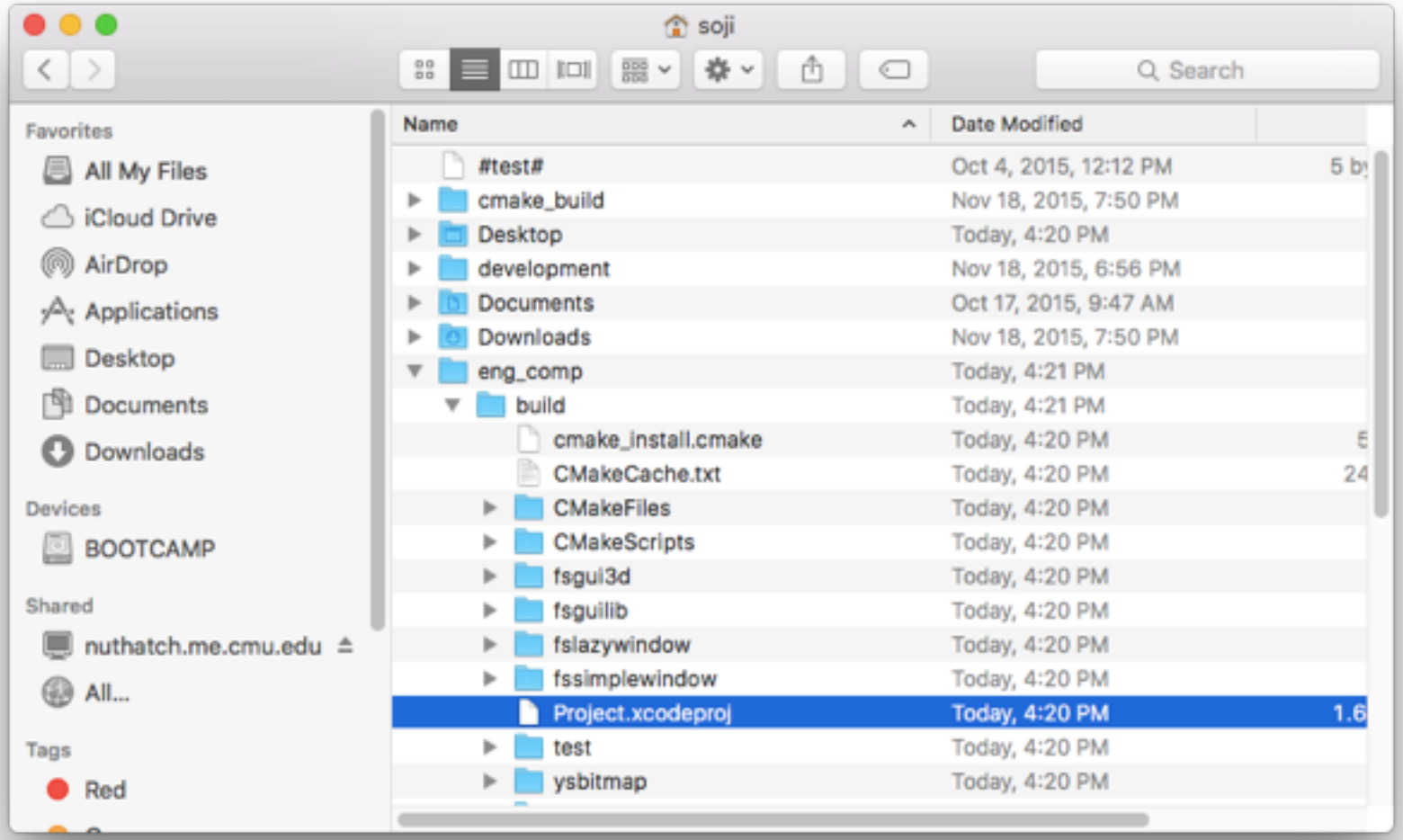


Building programs

- In Windows, locate and open Project.sln under the directory you specified as the build directory in CMake.
- Right-click “ysgebl64” in the Solution Explorer and build.
- (You can build everything, but you don’t have to wait for building tests and samples.)
- Then right-click ysgebl64 and Debug->New Instance
- By default, the program is built in the “Debug” mode. Change to “Release” mode for better performance.

Building programs

- In MacOSX, open Project.xcodeproj in the build directory, select ysgebl64 and build and run.



Adding your own project

- After you verify you checked out repository all right, just delete your build directory. You can re-create any time.

- Then check out your own repository from the server. (Replace hummingbird with your Andrew ID.)
  - In Windows

```
E:\>cd /D "%USERPROFILE%"

C:\Users\soji>cd eng_comp\src

C:\Users\soji\eng_comp\src>svn checkout https://ramennoodle/svn/teaching/24783_S16/students/hummingbird hummingbird
Checked out revision 14.
```

  - In MacOSX

Replace with your Andrew ID

```
[~] % cd ~/eng_comp/src
[~/eng_comp/src] % svn checkout https://ramennoodle/svn/teaching/24783_S16/students/hummingbird hummingbird
Checked out revision 14.
```

Copying a template from public repository

- Copy, move, and export
- svn copy *source destination*
  - Copying a file or a directory within the same repository.
- svn move *source destination*
  - Moving a file or a directory within the same repository.
- svn export *source destination*
  - Copying a file or a directory from a repository to somewhere else (can be another repository.)

- In this case, a program template needs to be copied from the public repository to your repository. Use svn export.

Copying a fssimplewindow template

- In Windows

```
C:\Users\soji\eng_comp\src>cd hummingbird

C:\Users\soji\eng_comp\src\hummingbird>svn export ..\public\src\fssimplewindow\template my_first_cmake_program
A    my_first_cmake_program\CMakeLists.txt
A    my_first_cmake_program\main.cpp
Export complete.

C:\Users\soji\eng_comp\src\hummingbird>svn add my_first_cmake_program
A         my_first_cmake_program
A         my_first_cmake_program\CMakeLists.txt
A         my_first_cmake_program\main.cpp

C:\Users\soji\eng_comp\src\hummingbird>svn commit . -m "Added my first cmake program."
Adding         my_first_cmake_program
Adding         my_first_cmake_program\CMakeLists.txt
Adding         my_first_cmake_program\main.cpp
Transmitting file data ..
Committed revision 15.

C:\Users\soji\eng_comp\src\hummingbird>cd my_first_cmake_program

C:\Users\soji\eng_comp\src\hummingbird\my_first_cmake_program>notepad CMakeLists.txt
```

- In MacOSX

```
[~/eng_comp/src] % cd hummingbird/
[~/eng_comp/src/hummingbird] % svn export ../public/src/fssimplewindow/template my_first_cmake_program
A    my_first_cmake_program/main.cpp
A    my_first_cmake_program/CMakeLists.txt
Export complete.
[~/eng_comp/src/hummingbird] % svn add my_first_cmake_program/
A         my_first_cmake_program
A         my_first_cmake_program/CMakeLists.txt
A         my_first_cmake_program/main.cpp
[~/eng_comp/src/hummingbird] % svn commit -m "Added my first cmake program"
Adding         my_first_cmake_program
```

```
[~/eng_comp/src/hummingbird] % cd my_first_cmake_program/
[~/eng_comp/src/hummingbird/my_first_cmake_program] % open CMakeLists.txt
```

- After svn add and svn commit, your files are stored in the course SVN server. You can check out from other PCs and work anywhere. After making changes, make sure you commit before logging off.
- If you make modifications on the same file from the two different locations, you may have to merge them together manually (which can be tedious.)

Giving a name to your program and add a top-level CMakeLists.txt

- Then, in the text editor, change:

  set(TARGET_NAME fssimplewindow_template)

  To

  set(TARGET_NAME my_first_cmake_program)

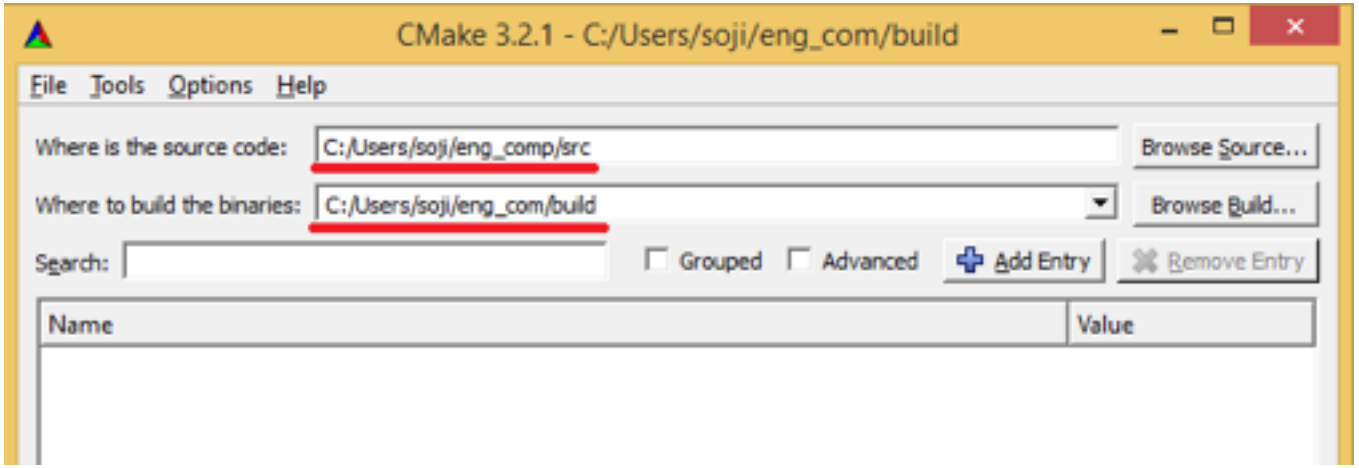
- Also add a CMakeLists.txt file of the following content in the (user directory)/eng_comp/src/CMakeLists.txt

```
cmake_minimum_required(VERSION 3.0)
include_directories(public/src/imported/include)
add_subdirectory(public/src)
add_subdirectory(hummingbird/my_first_cmake_program)
```

Cmake again

- This time, the source directory should be:
  (User directory)/eng_comp/src

- Make sure you delete previous build directory.
- Also close Xcode or Visual Studio before running Cmake.



Build my_first_cmake_program

- Open Project.sln file under your build directory.
- Build and run (or Debug) my_first_cmake_program.
- You don’t have to build other programs.

- From now on, do <u>NOT</u> add source files from XCode or Visual Studio.
- CMake will manage your source files.
- When you add a source file, do the following
  1. Create a file.
  2. Add the file to the project by editing CMakeLists.txt
  3. Add the file to the repository by:
     svn add (new_source_file_name)
  4. (Optinally) commit the file to the repository by:
     svn commit . -m "Added new source“
  5. Run cmake_gui (or cmake from command line).
  6. Then build.

Running CMake from command

In Windows

1. Cd or pushd to the build directory, and type:
   cmake ../src -G "Visual Studio 12 Win64"
2. To build, type:
   msbuild Project.sln /m /target:my_first_cmake_program

In MacOSX

1. Cd or pushd to the build directory, and type:
2. To build, type:

About CMakeCache.txt

- When you want to change build setting, typically when you want to change the generator (like from 32 bit to 64 bit), you may get an error from CMake.
- When it happens, delete a file called CMakeCache.txt in the build directory and try again.

Working from multiple computers

- Make sure to commit your changes after working on one PC.
- Or, if you totally messed up and want to discard all changes, change directory to the working copy and type:

  ```
  svn revert . --depth infinity
  ```

- You can also delete (I rather recommend to rename) sources and check out fresh from the repository. But, make sure you really committed your files.
- In another PC, if you have already checked out from the same repository, do update before start working. Just change directory to the working copy, and type:

  ```
  svn update
  ```

Minimum CMakeLists.txt

- If your program does not link any libraries other than C/C++ standard libraries, and if the program consists of only one source file called main.cpp, all you need is:

```
add_executable(minimum_cmake_sample main.cpp)
```

- It tells that the program called minimum_cmake_sample is built from main.cpp
- If the program links to a library called fssimplewindow, you need one more line:

```
target_link_libraries(minimum_cmake_sample fssimplewindow)
```

- Also you want to say

```
add_executable(minimum_cmake_sample MACOSX_BUNDLE main.cpp)
```

to support build for MacOSX application.

To use C++11 features

- Things get a bit complex if you want to use C++11 features, you need to add compiler-specific flags for clang and gcc.

```
if(MSVC)
elseif(APPLE)
        set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -std=c++11")
else()
        set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -std=c++0x")
endif()
```

- If your gcc supports –std=c++11, it can be as simple as:

```
if(NOT MSVC)
        set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -std=c++11")
endif()
```

-

- To be realistic and more general, you need to fine-tune some more settings.
- CMakeLists.txt in the templates takes care of these fine-tuning.