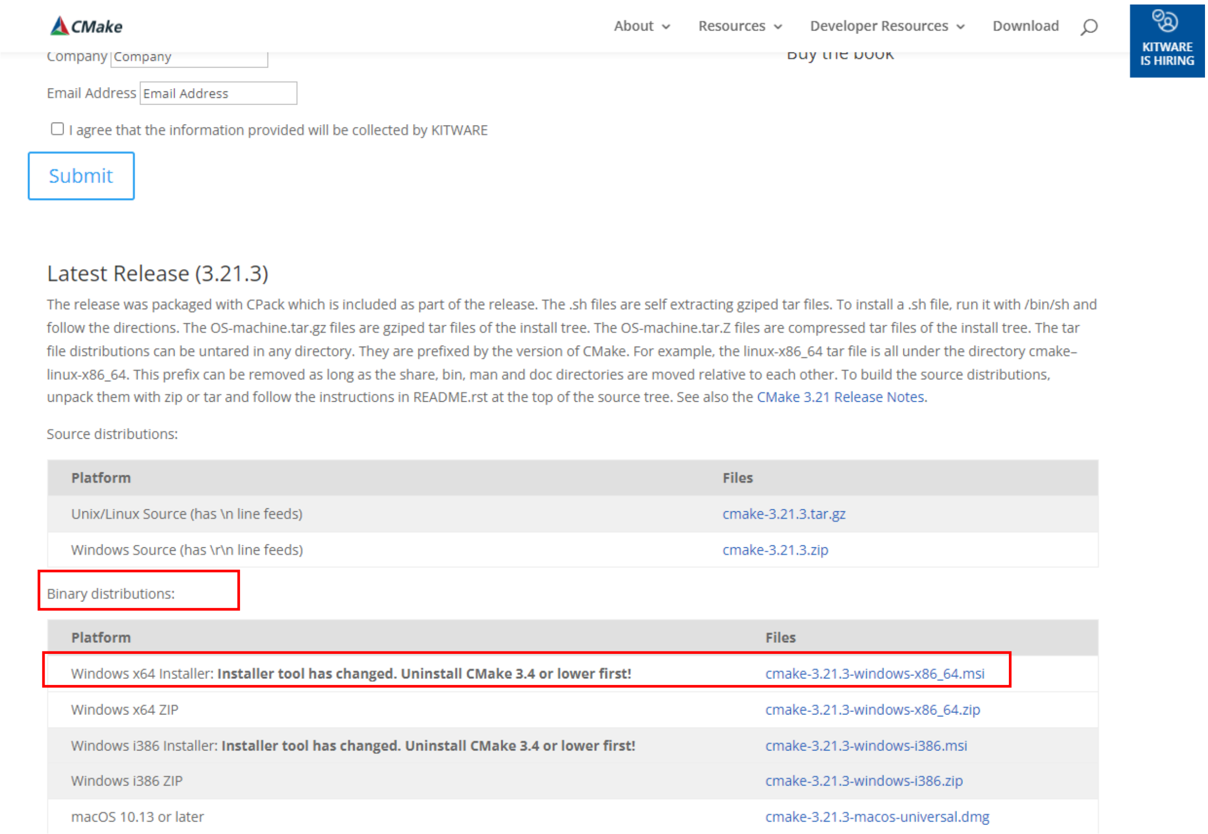
How to compile solution with CMake?

Step 1: download CMake

https://cmake.org/download/

select and download “Binary distributions - platform – Windows x64 Installer”



Company Company

Email Address Email Address

□ I agree that the information provided will be collected by KITWARE

Submit

Latest Release (3.21.3)

The release was packaged with CPack which is included as part of the release. The .sh files are self extracting gziped tar files. To install a .sh file, run it with /bin/sh and follow the directions. The OS-machine.tar.gz files are gziped tar files of the install tree. The OS-machine.tar.Z files are compressed tar files of the install tree. The tar file distributions can be untared in any directory. They are prefixed by the version of CMake. For example, the linux-x86_64 tar file is all under the directory cmake–linux-x86_64. This prefix can be removed as long as the share, bin, man and doc directories are moved relative to each other. To build the source distributions, unpack them with zip or tar and follow the instructions in README.rst at the top of the source tree. See also the CMake 3.21 Release Notes.

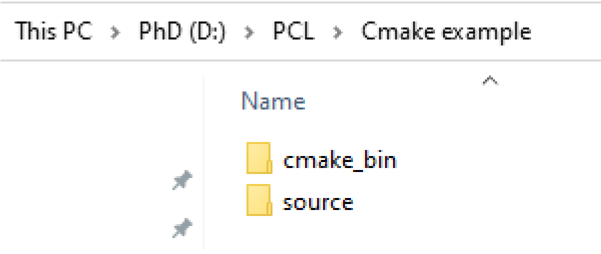
Source distributions:

Platform	Files
Unix/Linux Source (has \n line feeds)	cmake-3.21.3.tar.gz
Windows Source (has \r\n line feeds)	cmake-3.21.3.zip

Binary distributions:

Platform	Files
Windows x64 Installer: **Installer tool has changed. Uninstall CMake 3.4 or lower first!**	cmake-3.21.3-windows-x86_64.msi
Windows x64 ZIP	cmake-3.21.3-windows-x86_64.zip
Windows i386 Installer: **Installer tool has changed. Uninstall CMake 3.4 or lower first!**	cmake-3.21.3-windows-i386.msi
Windows i386 ZIP	cmake-3.21.3-windows-i386.zip
macOS 10.13 or later	cmake-3.21.3-macos-universal.dmg

Step 2: create two folders under your path



This PC › PhD (D:) › PCL › Cmake example

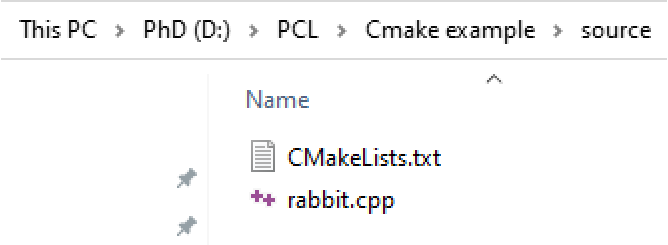
Name

cmake_bin

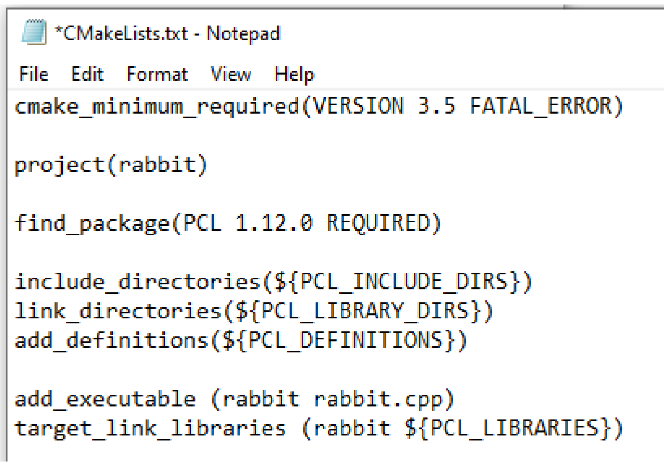
source

1) The first folder named “source”
2) The second folder named “cmake_bin”

Step 3: add name.cpp file and CMakeLists.txt file in source



1) Here, we use rabbit.cpp as the C++ example code
2) Create a text file named “CMakeLists.txt”
3) Write settings in CMakeLists.txt: check the words in red. Add_executable’s names and target_link_libraries’ name must be the same.

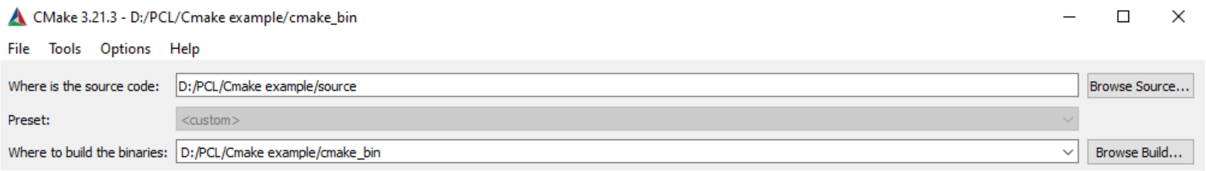


```
cmake_minimum_required(VERSION 3.5 FATAL_ERROR)

project(rabbit)

find_package(PCL 1.12.0 REQUIRED)

include_directories(${PCL_INCLUDE_DIRS})
link_directories(${PCL_LIBRARY_DIRS})
add_definitions(${PCL_DEFINITIONS})

add_executable (rabbit rabbit.cpp)
target_link_libraries (rabbit ${PCL_LIBRARIES})
```

```
cmake_minimum_required(VERSION 3.5 FATAL_ERROR)
project(rabbit)
find_package(PCL 1.12.0 REQUIRED)
include_directories(${PCL_INCLUDE_DIRS})
link_directories(${PCL_LIBRARY_DIRS})
add_definitions(${PCL_DEFINITIONS})
add_executable (rabbit rabbit.cpp)
target_link_libraries (rabbit ${PCL_LIBRARIES})
```

Step 4: Compile with CMake

1) Open CMake by “run as administrator”
2) Select “source code” path and “where to build the binaries” path



Where is the source code: D:/PCL/Cmake example/source

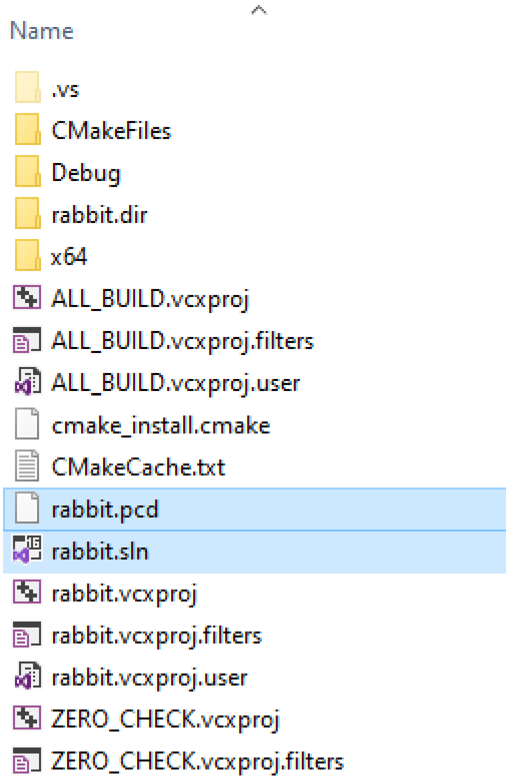
Preset: <custom>

Where to build the binaries: D:/PCL/Cmake example/cmake_bin

3) Click “configure”, select SV 2019 and X64
4) Click “generate”

5) If it shows “configuration down” and “generation down”, then close CMake

6) The cmake_bin folder should look like this:



Name

.vs
CMakeFiles
Debug
rabbit.dir
x64
ALL_BUILD.vcxproj
ALL_BUILD.vcxproj.filters
ALL_BUILD.vcxproj.user
cmake_install.cmake
CMakeCache.txt
rabbit.pcd
rabbit.sln
rabbit.vcxproj
rabbit.vcxproj.filters
rabbit.vcxproj.user
ZERO_CHECK.vcxproj
ZERO_CHECK.vcxproj.filters

7) Make sure you put point cloud file under this folder: rabbit.pcd

Step 5: Run and Check

8) Open “rabbit.sln” file, “ctrl+F5” to debug the code

Reference:

https://pointclouds.org/documentation/tutorials/using_pcl_pcl_config.html#using-pcl-pcl-config