<Open3D C++ Build from source with CMake>

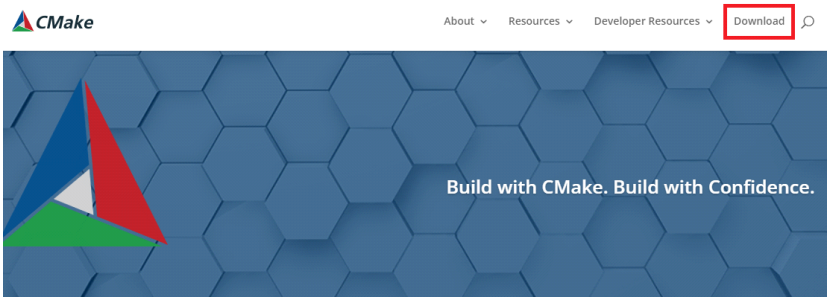
2017013100 컴퓨터과학과 유성민 (증강지능 연구실원)

※ 설정 환경

- 설치할 버전 : Open3D 0.13.0
- Windows 10 (64bit) : Visual Studio 2019+ (C++14 Compiler)
- CMake : 버전 3.18+
- Git (선택 요소)
- Nvidia CUDA Toolkit 11.0 (선택 요소)

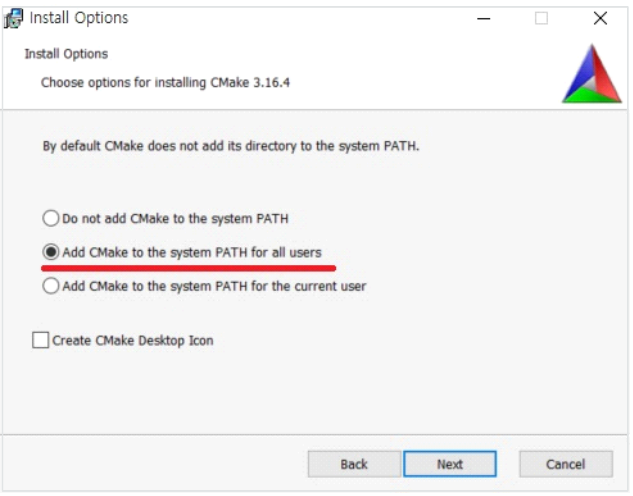
1. CMake 설치

웹페이지 링크 : https://cmake.org/



CMake 홈페이지에 접속해 Download 카테고리 클릭

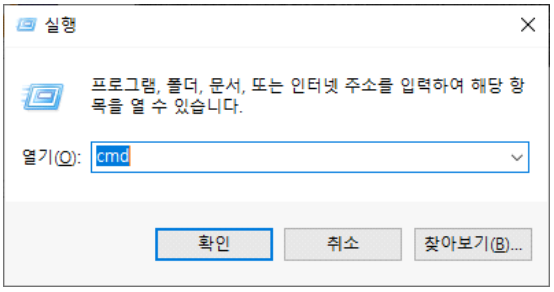
Windows x64 Installer (msi 버전) 클릭 후 다운로드 및 설치



○

설치 중에 Add CMake to the system PATH for all users 선택

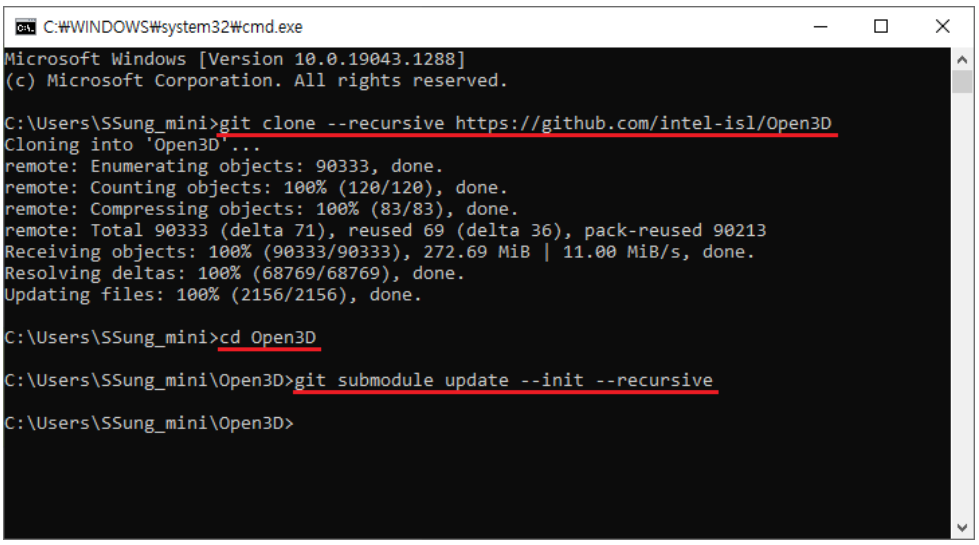
2. Open3D Source 설치



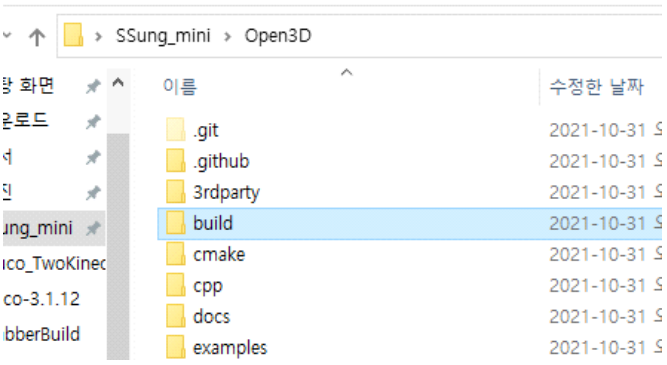
명령 프롬프트(cmd) 실행

```
git clone --recursive https://github.com/intel-isl/Open3D
# 하위 모델을 수동적으로 업데이트 하는 과정
cd Open3D
git submodule update --init --recursive
```

명령 프롬프트에 위 명령어를 입력해 Cloning 수행



submodule update 후 아무 로그가 뜨지 않는다면 에러 없이 완료한 것임



Cloning된 Open3D를 확인 후, 폴더 안에 "build" 폴더 생성

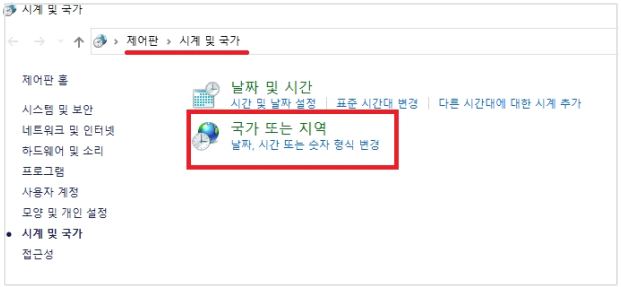
※ Git을 사용하지 않는다면 https://github.com/isl-org/Open3D 에 접속해 직접 코드를 다운로드함.

3. 빌드 전 사전 설정

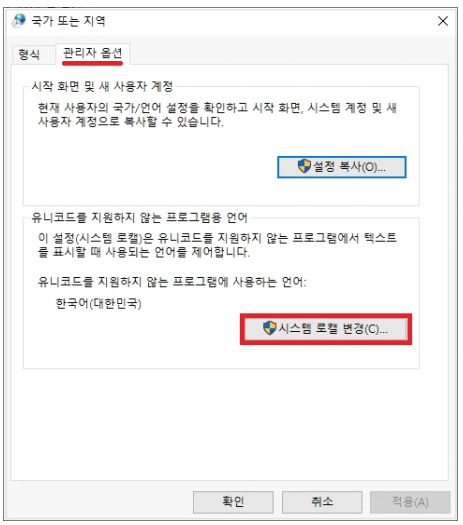
이 설정을 하지 않고 빌드를 수행하게 되면 비주얼 스튜디오에서 **Warning : C4819** 와 **ERROR : C2220**를 마주할 것이다.

warning C4819	현재 코드 페이지(949)에서 표시할 수 없는 문자가 파일에 들어 있습니다. 데이터가 손실되지 않게 하려면 해당 파일을 유니코드 형식으로 저장하십시오.
error C2220	다음 경고는 오류로 처리됩니다.

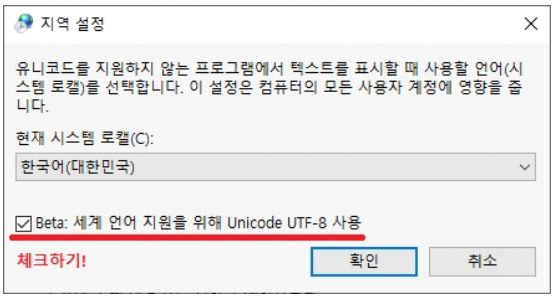
이 오류는 해외의 경우에는 잘 나타나지 않는다. 영문 언어의 OS가 아닌 영문 외 **한글 언어의 OS를 사용**하게 되면 UTF-8 유니코드 형식으로 저장하라는 Warning이지만, 이 수가 너무 많아 결국 컴파일러에서 Error로 처리하게 되어 빌드에 실패하게 된다. 그러므로 이 Error를 사전 방지하기 위해 아래와 같은 설정을 수행한다.



제어판 > 시계 및 국가 > 국가 또는 지역 클릭



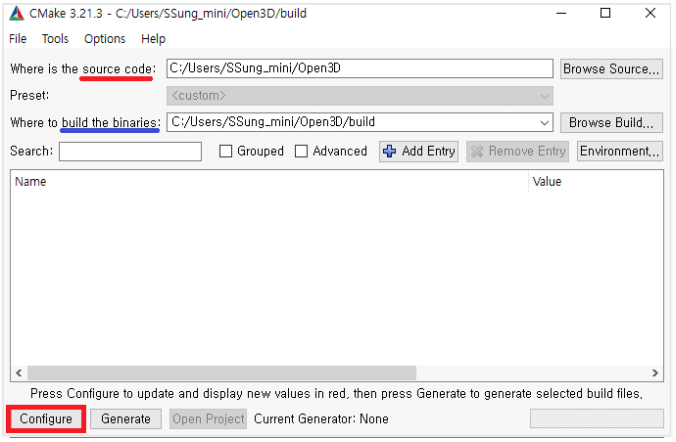
관리자 옵션 탭 > 시스템 로캘 변경 클릭



Unicode UTF-8 사용 체크 후 재부팅

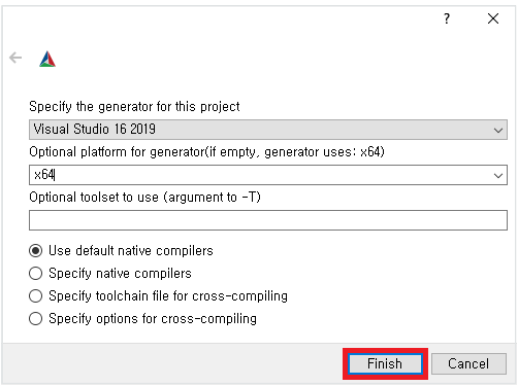
- Unicode UTF-8 사용 시 한글 오피스로 PDF 저장 시 오류가 나므로 모든 빌드 작업 후에 체크 해제를 권장

4. CMake (cmake-gui) 실행 (※ 관리자 권한으로 실행하기)

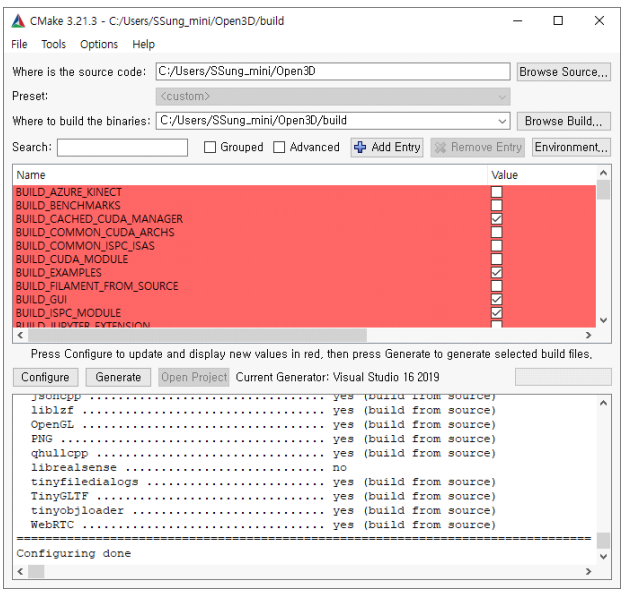


- Where is the source code : 기본 소스 코드가 들어있는 폴더 선택 **(Cloning한 Open3D 폴더)**
- Where to build the binaries : make할 폴더 선택 **(Open3D 폴더 내의 임의로 만든 Build 폴더)**

위와 같이 설정 후 Configure 클릭



위와 같이 설정 후 Finish 클릭



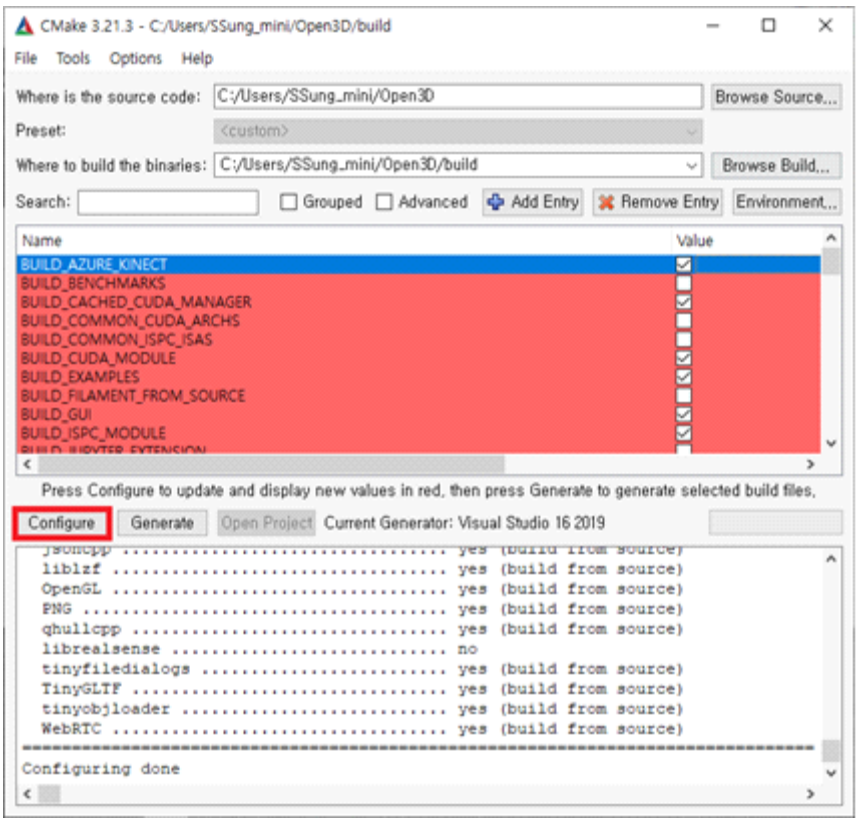
Configure이 정상적으로 완료되면 위와 같은 화면이 표시됨

빨간색으로 색칠된 항목은 **새롭게 업데이트된 항목**으로 확인하라는 표시

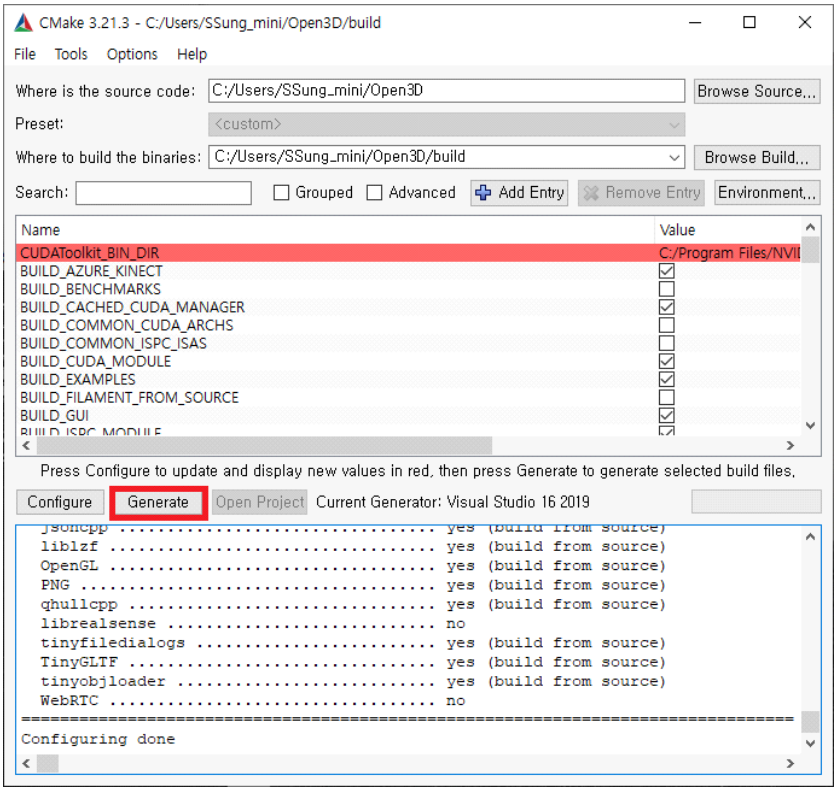
- 출력된 목록들 중에서 사용하는 것들은 **체크**, 빌드하지 않을 것들은 **체크를 해제**해줌.
- 빌드를 위해 아래의 **필수적으로 수정해야 할 목록들**을 확인 **(Search를 사용하면 편리함)**

수정해야할 목록 (ON = 체크 / OFF = 체크 해제)	설명
BUILD_PYTHON_MODULE = **OFF**	위 문서에서는 Python이 아닌 C++만을 위한 빌드이므로 PYTHON MODULE 설치를 하지 않음. Python과 관련된 것들이 체크되어있다면 체크 해제함
BUILD_SHARED_LIBS = **ON**	Shared Library 파일들을 생성하기 위함
BUILD_WEBRTC = **OFF**	BUILD_SHARED_LIBS와 BUILD_WEBRTC를 동시에 ON 하게 되면 에러 유발 (MSVC에서 지원하지 않음)
STATIC_WINDOWS_RUNTIME = **OFF**	체크 후 Configure 시 에러 발생
CMAKE_INSTALL_PREFIX = **"C:₩Open3D"**	BUILD 후 LIB, DLL 등 설정 파일들이 저장되는 장소. 기본 설정인 "C:/Program Files/Open3D"에 저장 시 관리자 권한 접근으로 인해 빌드에 실패할 수 있음. 예시로 "C:/Open3D"에 설정하였으며, 원하는 경로로 설정하면 됨.

- 추가적으로 Open3D에서 **Azure Kinect나 Intel Realsense, CUDA_MODULE 등을 사용**하려 한다면 **체크**해주어야 하고, 사전에 드라이버나 SDK들을 설치해주는 것이 좋음
- 이 문서에서는 예시로 **BUILD_AZURE_KINECT와 BUILD_CUDA_MODULE 을 추가로 체크**하여 작성함



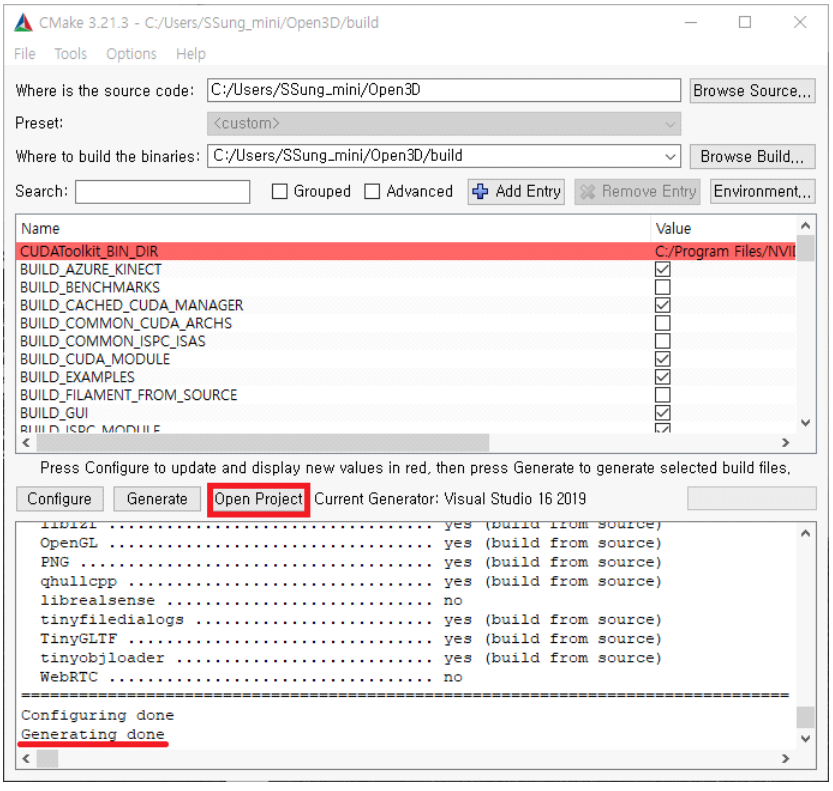
모든 설정이 완료되었다면 **Configure 버튼**을 한번 더 클릭함
(설정한 것들을 적용시키는 단계)



추가적으로 빨간 리스트가 있다면 확인하고,

수정해야 할 항목이 없고 log에 "Configuring done" 이라고 표시가 되었다면 Generate를 클릭함

Generate : Configure한 소스 설정들을 적용하여 솔루션(프로젝트) 파일을 생성시킴

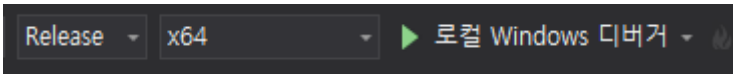


Generate 후 log에 "Generationg done" 이라고 표시되면 정상적으로 Generating된 것임

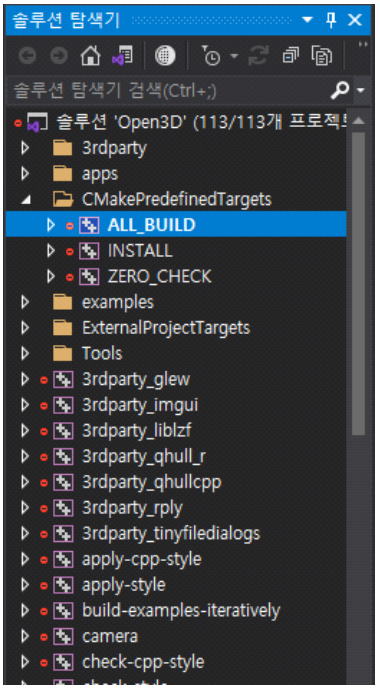
마무리되었다면 Open Project를 클릭해 솔루션 파일을 관리자 권한으로 열어줌

※ CMake를 관리자 권한으로 실행하였다면, Open Project 클릭 시 **자동으로 관리자 권한으로 VS가 열림**

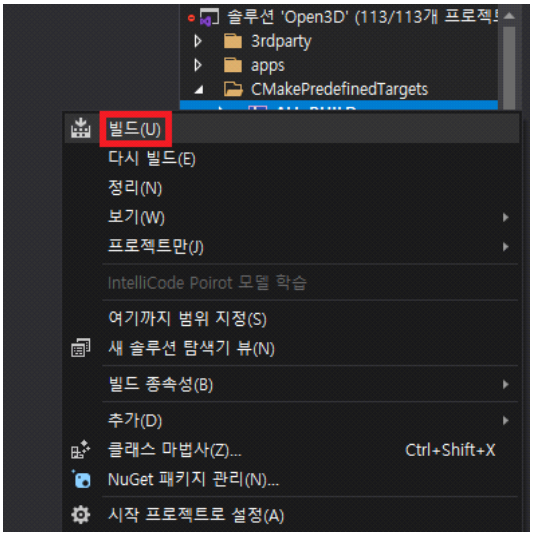
5. Visual Studio를 이용하여 빌드하기



비주얼 스튜디오로 솔루션이 정상적으로 실행되었다면 **Release – x64**로 변경

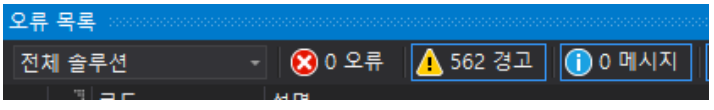


솔루션 Open3D > CMakePredefinedTargets 폴더의 **ALL_BUILD**를 **시작 프로그램으로 설정**해줌

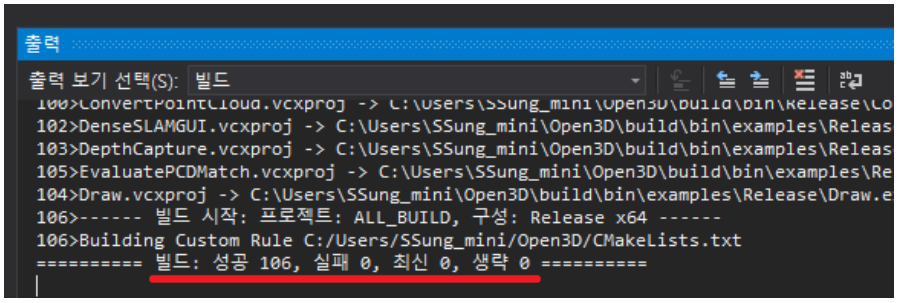


ALL_BUILD를 오른쪽 클릭하여 **빌드**를 클릭

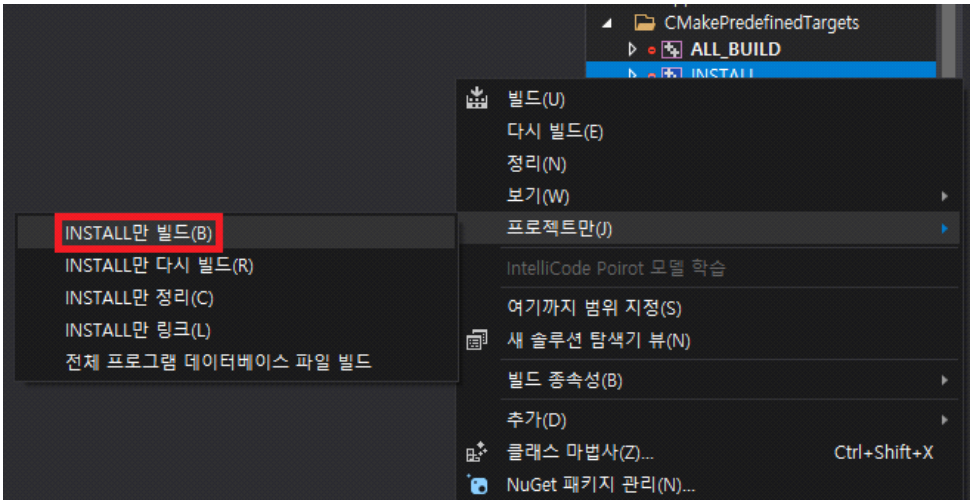
- 빌드는 컴퓨터의 성능에 따라 **평균 20분 ~ 1시간**정도 소요됨



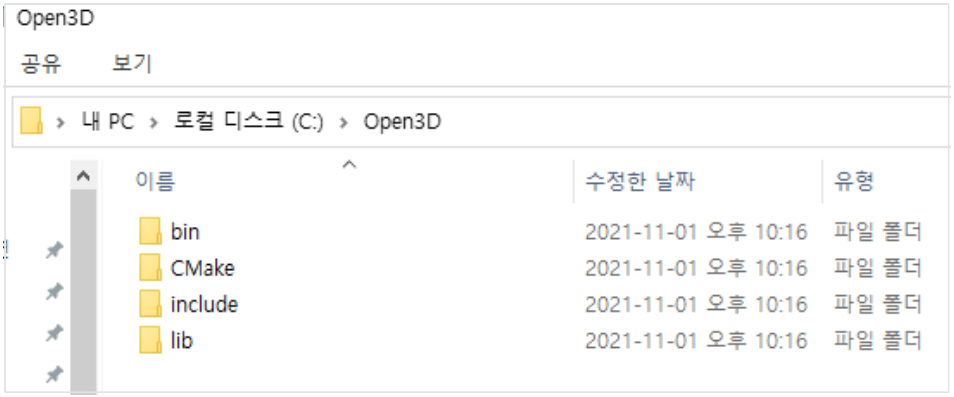
경고는 아무리 떠도 상관없지만, **오류는 0을 유지**해야함



빌드가 마무리되었을 때, 실패가 하나도 없어야하며, 성공만 나타나야 정상적으로 빌드된 것



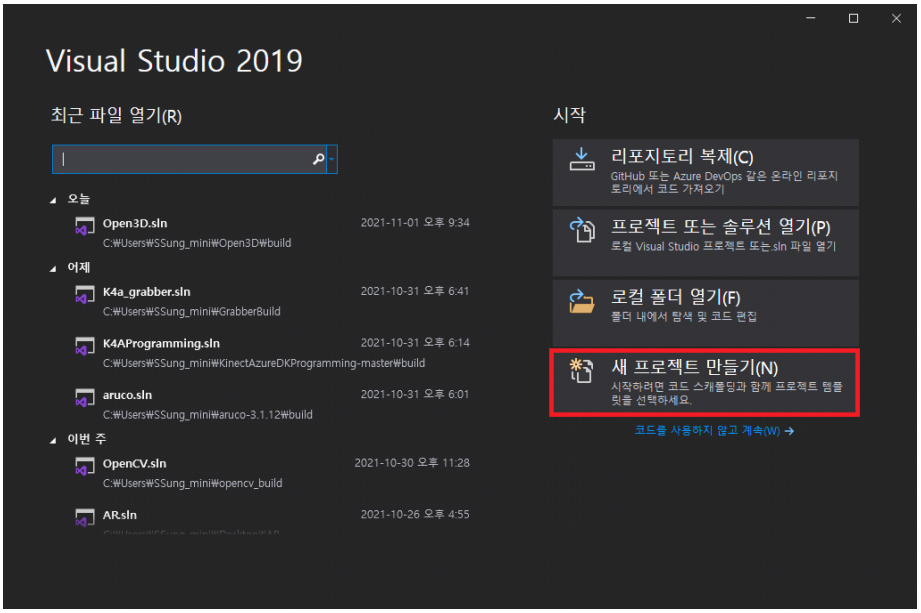
빌드를 성공적으로 마쳤다면 ALL_BUILD 아래에 있는 **INSTALL** 오른쪽 클릭
프로젝트만 > INSTALL만 빌드 클릭



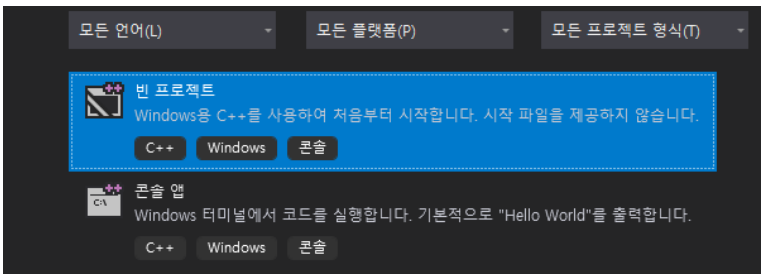
INSTALL 또한 잘 완료되었다면, CMake에서 설정했던 **CMAKE_INSTALL_PREFIX의 경로**에 bin, include, lib, CMake 파일들이 저장되어 있는 것을 확인할 수 있음

6. 임의의 프로젝트에 적용 (Find Pre-Installed Open3D Package)

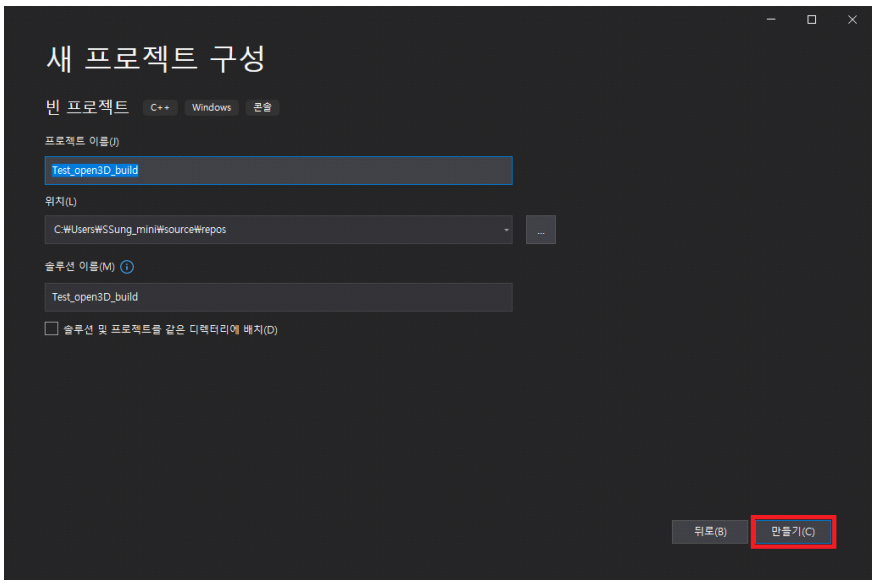
- 적용하는 방법은 OpenCV를 적용하는 방법과 유사함
- 본 적용법은 Visual Studio로 생성한 Solution 안에 Open3D 파일들을 추가해 연결하는 방법을 사용함



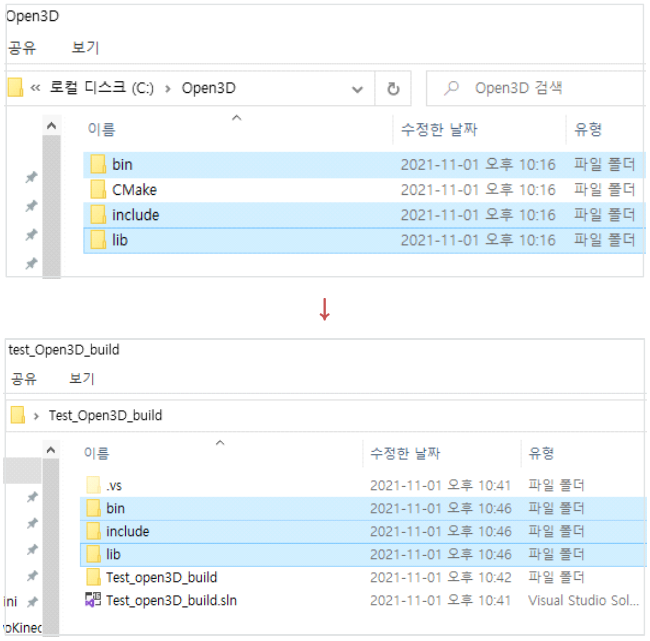
임의의 프로젝트를 생성하기 위해 Visual Studio를 실행해 '**새 프로젝트 만들기**' 클릭



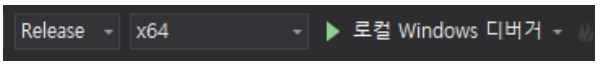
'**빈 프로젝트**' 선택



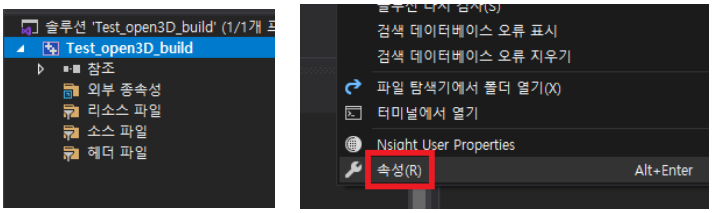
원하는 이름으로 작성 후 프로젝트를 생성해줌



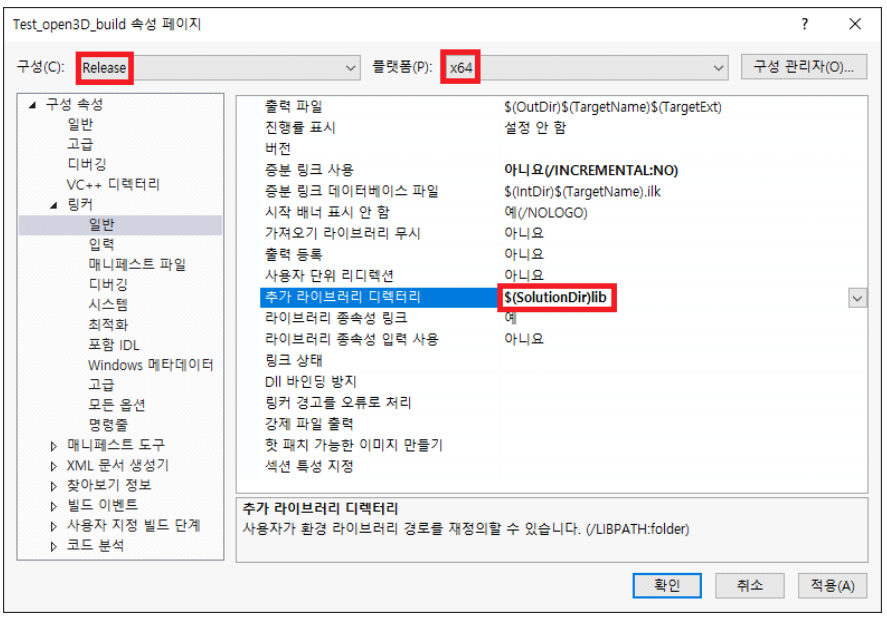
Build했던 Open3D 폴더의 **'bin, include, lib' 폴더를 복사**해 프로젝트 폴더에 추가해줌



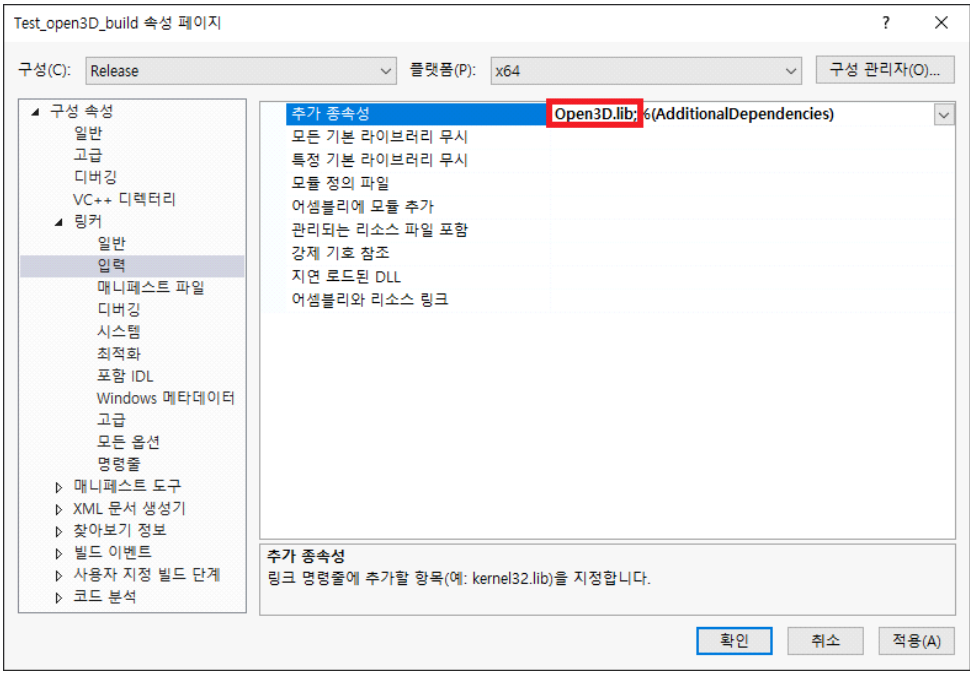
Open3D에서는 모든 설정을 Release – x64**로 세팅**하여 사용함



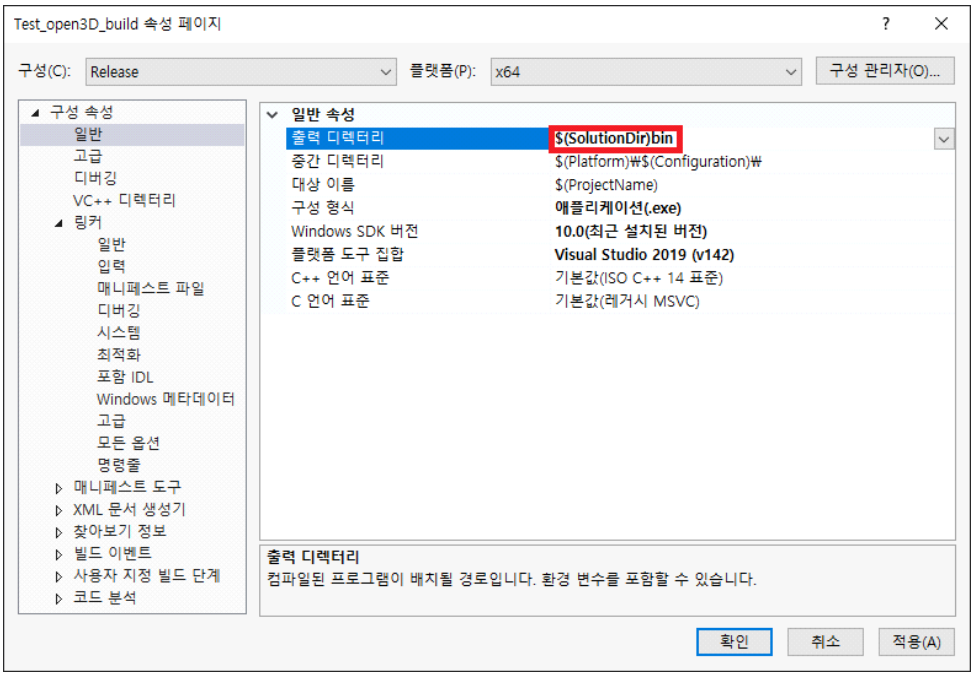
생성된 프로젝트를 **오른쪽 클릭 > 속성** 클릭



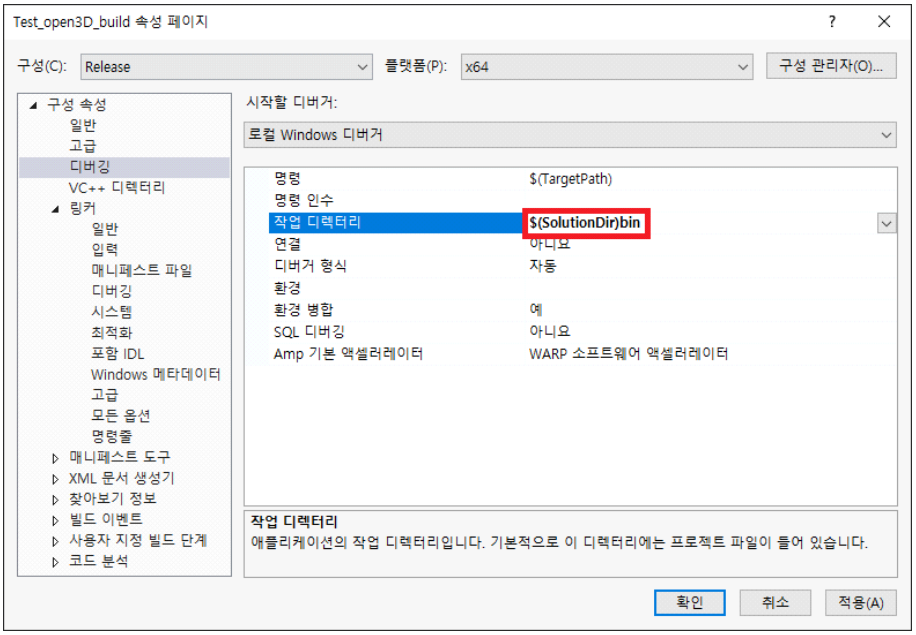
링커 > 일반 > 추가 라이브러리 디렉터리 : $(SolutionDir)lib



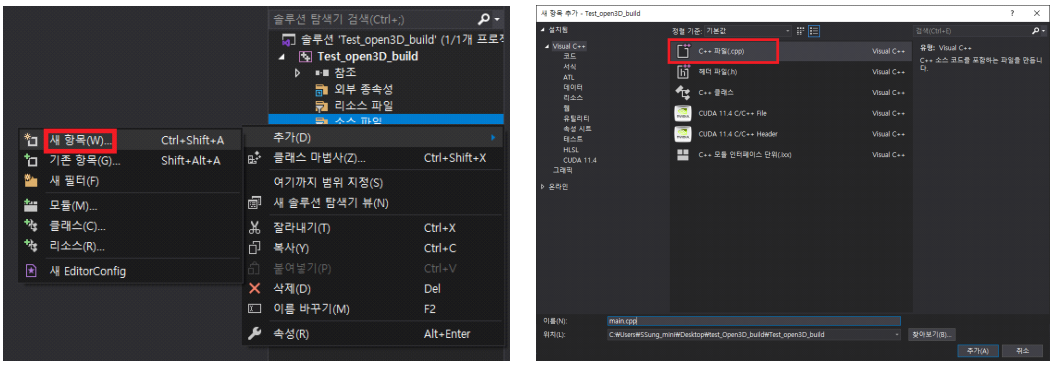
링커 > 일반 > 추가 종속성 : Open3D.lib 추가



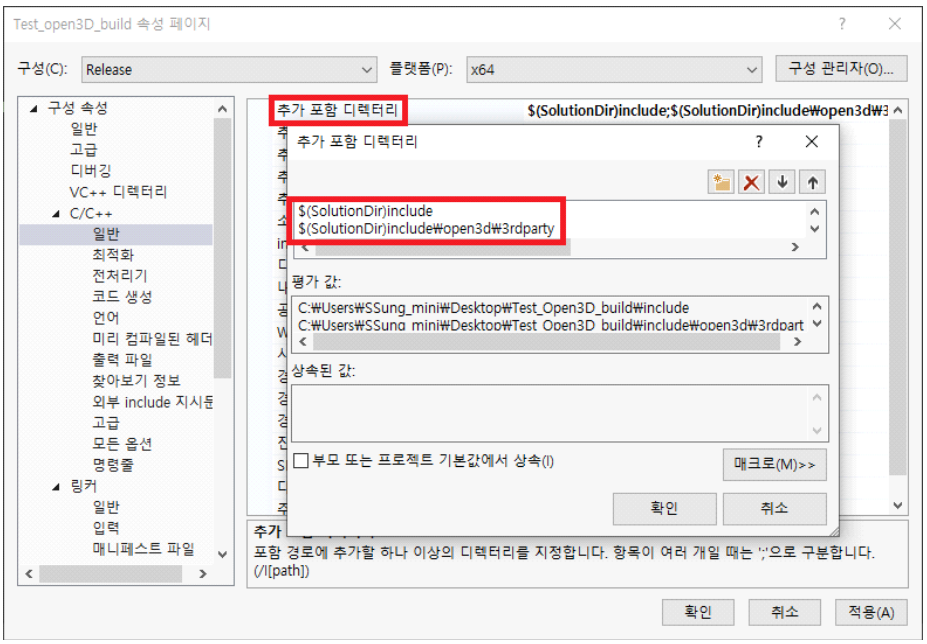
구성속성 > 일반 > 출력 디렉터리 : $(SolutionDir)bin



구성 속성 > 디버깅 > 작업 디렉터리 : $(SolutionDir)bin



프로젝트에서 C++ 파일을 생성함



C++ 파일 생성 후에 다시 프로젝트 속성에 진입해

C/C++ > 일반 > 추가 포함 디렉터리 : $(SolutionDir)include; $(SolutionDir)include₩open3d₩3rdparty

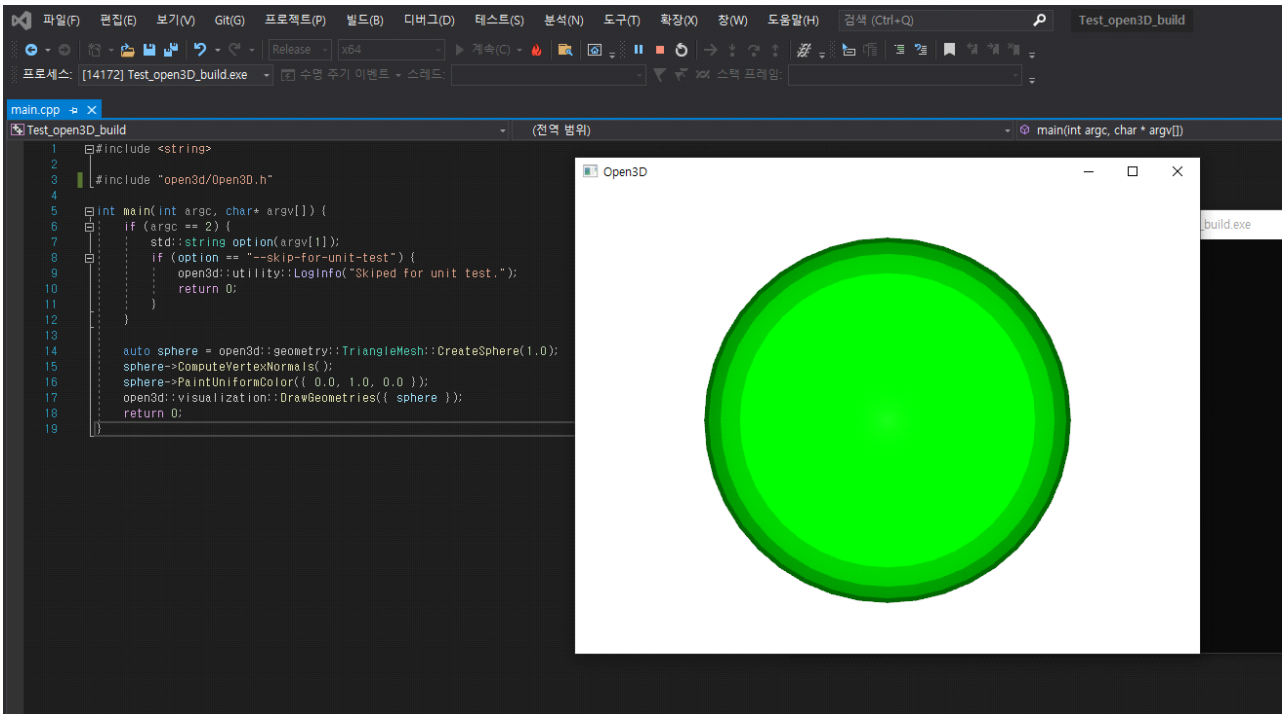
```cpp
#include <string>

#include "open3d/Open3D.h"

int main(int argc, char* argv[]) {
    if (argc == 2) {
        std::string option(argv[1]);
        if (option == "--skip-for-unit-test") {
            open3d::utility::LogInfo("Skiped for unit test.");
            return 0;
        }
    }

    auto sphere = open3d::geometry::TriangleMesh::CreateSphere(1.0);
    sphere->ComputeVertexNormals();
    sphere->PaintUniformColor({ 0.0, 1.0, 0.0 });
    open3d::visualization::DrawGeometries({ sphere });
    return 0;
}
```

예제 코드 (출처 : open3d-cmake-find-package/Draw.cpp)



정상 실행 완료된 모습

REFERENCES

Open3D Docs : http://www.open3d.org/docs/release/

Open3D github : https://github.com/isl-org/Open3D

Open3D Build from source : http://www.open3d.org/docs/release/compilation.html

CMake Config setting : https://github.com/isl-org/Open3D/issues/3589

C2220, C4819 Error : https://github.com/dotnet/aspnetcore/issues/24908

C2220 Error : https://docs.microsoft.com/ko-kr/previous-versions/ksk5k0ta(v=vs.120)?redirectedfrom=MSDN

Open3D dynamic setting : https://github.com/isl-org/Open3D/issues/2717

tutorial simple code (open3d-cmake-find-package) : https://github.com/isl-org/open3d-cmake-find-package