## Instructions for compiling ISOmodel using Visual Studio 2022 / VS 2022 community edition

If you plan to work from the publicly available, open source, ISOmodel git repository and keep your changes/additions open source, the terms of use allow use of VS 2022 community edition even if you are a commercial business or enterprise. If you are a commercial company and plan on keeping those changes within your company or releasing in closed source software, you cannot use VS 2022 community edition and must have a licensed version VS 2022 Pro.

In the setup process it’s best to install CMake first so that the Visual Studio environment can find CMake during its own install.

## Install CMake >=3.12.3

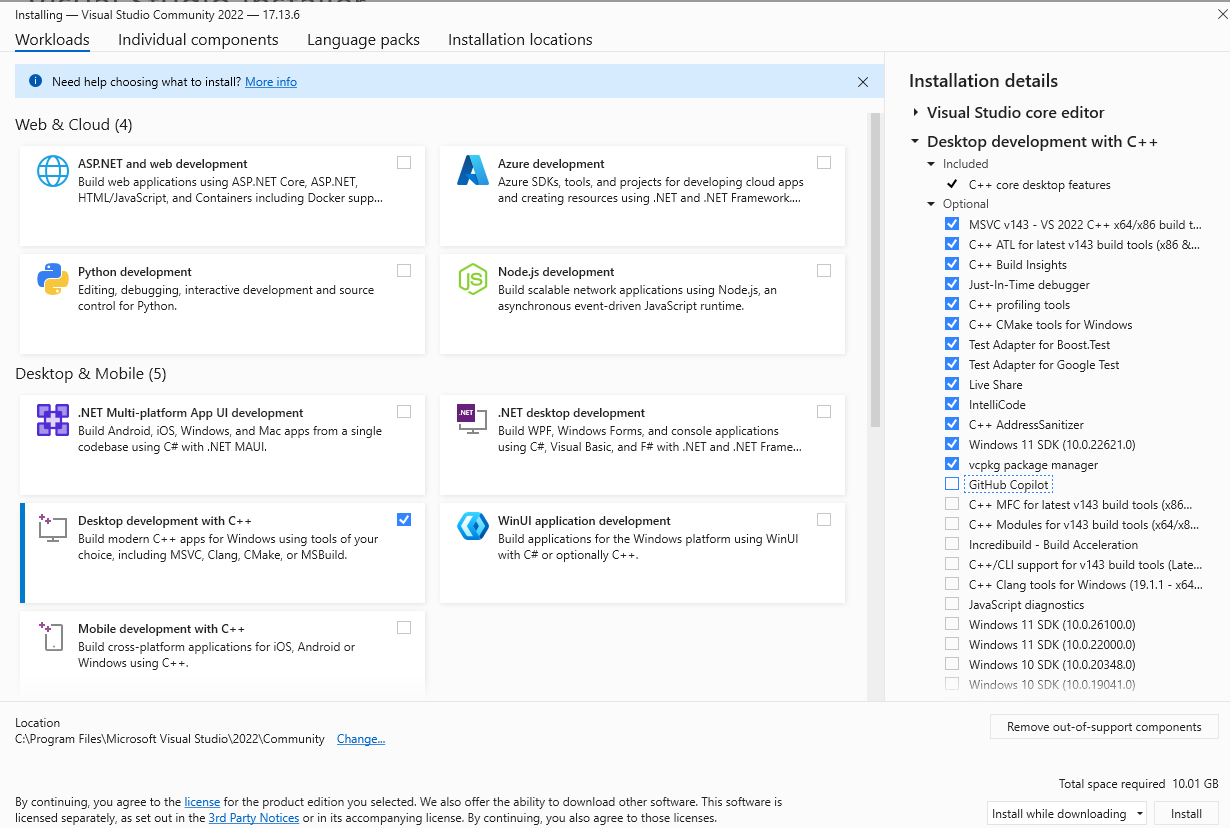
<https://github.com/Kitware/CMake/releases/download/v4.0.1/cmake-4.0.1-windows-x86_64.msi>

Install with the option “Add CMake to the PATH environment variable” checked.

This will prompt for admin login

## Download VS 2022/ VS 2022 Community Edition and begin installing

Select Desktop development with C++ and use default settings. You might want to uncheck GitHub Copilot if you don’t want more Copilot stuff. Here is what my install looks like with GitHub Copilot unchecked.



## Install Boost >=1.88 prebuilt binaries for msvc 14.3-64

(Visual Studio 2022 Version > 17.10 is actually msvc 14.4-64 but things have been set to work with the newer updates to VS 2022)

<https://sourceforge.net/projects/boost/files/boost-binaries/1.88.0/boost_1_88_0-msvc-14.3-64.exe/download>

You likely will get a warning from Microsoft that the software is suspect If you downloaded it direct from sourceforge it should be safe to override the warning and install.

Install to a simple easy to find directory like C:\local\boost\_1\_88\_0

Graphical user interface, text, application

AI-generated content may be incorrect.

## Install Google Test >= 1.16.0

I recommend you build googletest yourself with the latest version right from github

## Clone the googletest repository and make googletest

If you use the command line use the command

git clone <https://github.com/google/googletest.git>

Or, use the URL above in your favorite git GUI to clone the repository. In my case it was cloned to the c:\git\googletest directory. Once you have it cloned, open a command line / terminal prompt and cd into the main directory of the googletest repository. Make a build directory and run cmake from that directory. You might want to skip making Google Mock since we are not using it by adding the -DBUILD\_GMOCK=off to the cmake command line.

On my PC it looks like the following:

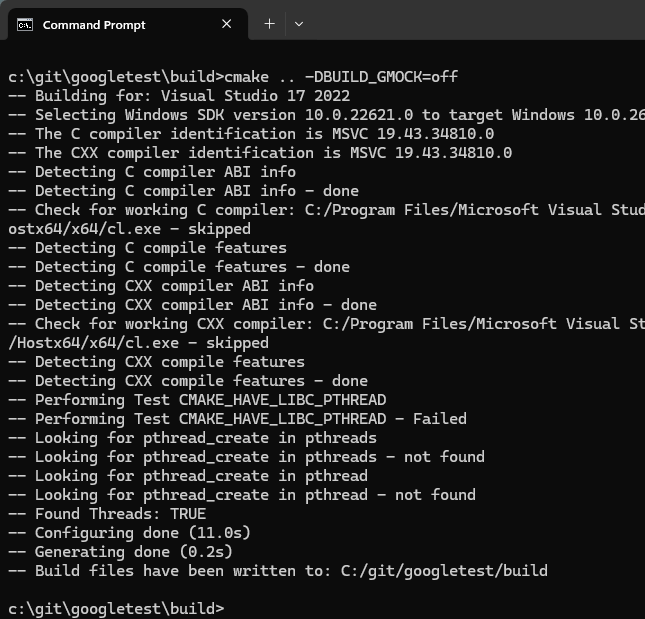
cd C:\git\googletest

C:\git\googletest>mkdir build

C:\git\googletest>cd build

C:\git\googletest\build>cmake .. -DBUILD\_GMOCK=OFF

Output should look something like this if it builds correctly

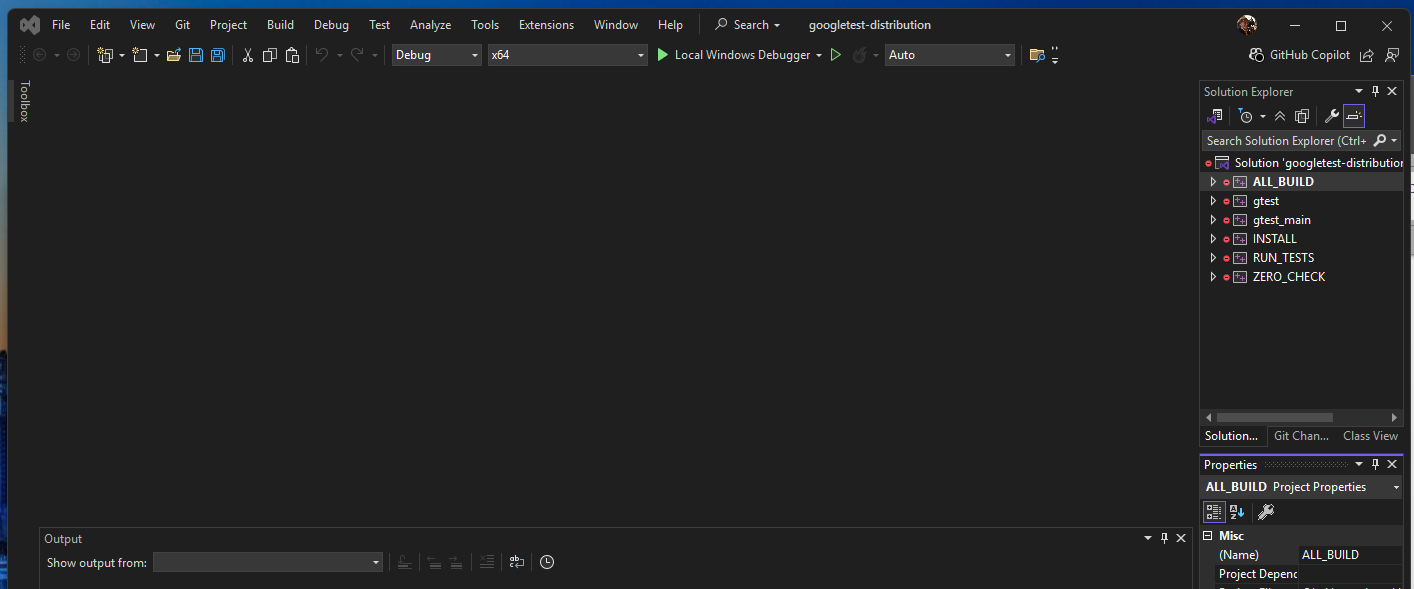


In the build directory you should now have a “googletest-distribution.sln” file along with some Cmake stuff and some .vcxproj files. The .sln is a visual studio project file that you can use to make/build googletest with visual studio 2022

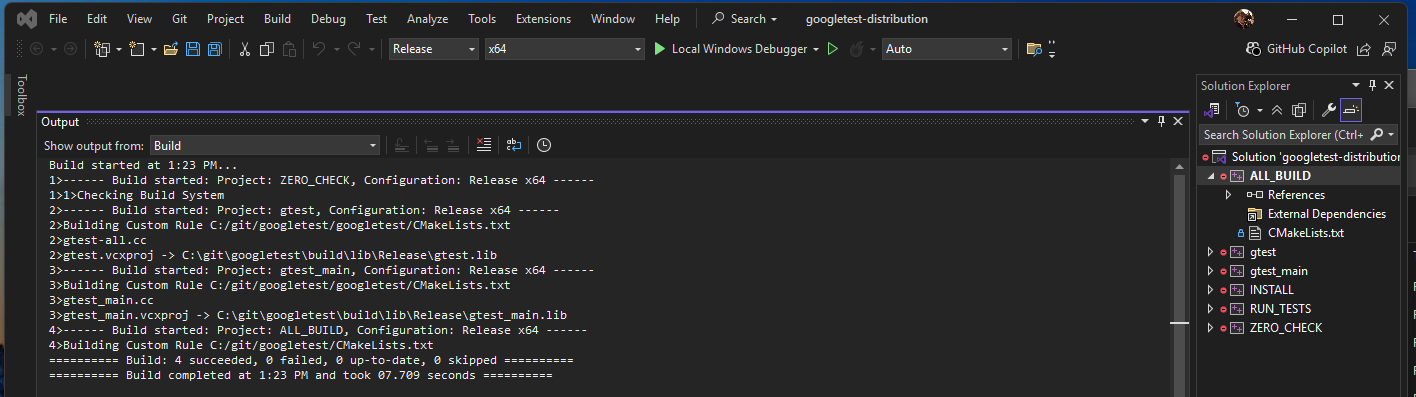
## Make googletest

You want to load the Visual Studio project into VS2022 by either double clicking on the googletest-distribution.sln file or starting VS2022 and choosing to open a project and finding the same file through the file browser.

Once open, you should have something that looks like the following:



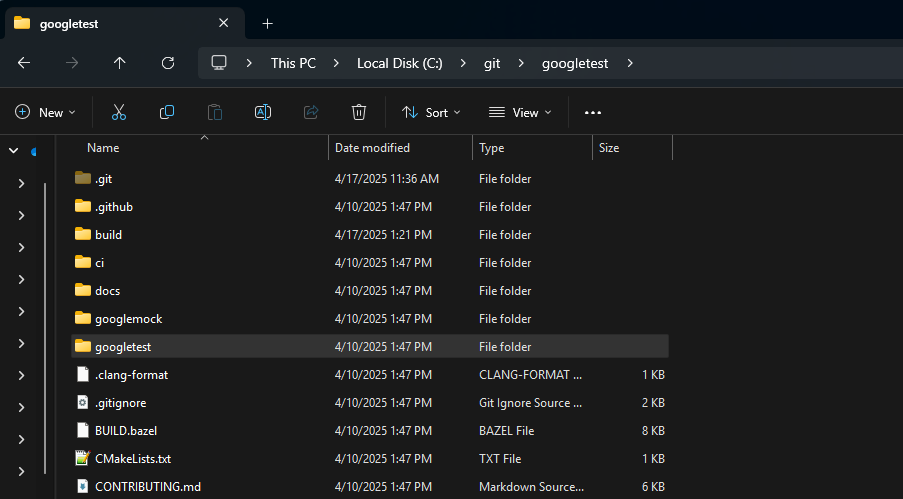
Change the setting to release and right-click on ALL\_BUILD and choose Build to start the build process. If it completes correctly the output should look something like this:



This should have created the files “gtest.lib” and “gtest\_main.lib” in the build\lib\Release directory

## Installing Googletest

Next we should make it easier to link to the googletest include files and libraries by copying things to a parallel directory to boost. Insite the googletest directory is another googletest directory (parallel to the build directory you just made).



Copy that directory to c:\local\googletest

On my computer that means copy C:\git\googletest\googletest to C:\local\googletest

Within the C:\local\googltest directory, create a new directory for the libs and copy over the gtest.lib and gtest-main.lib that were created in the release folder by the build you just did

mkdir C:\local\googltest\lib

Copy the following files into the new lib directory you just made

googletest/build/lib/Release/gtest.lib

googletest/build/lib/Release/gtest-main.lib

## Compiling isomodel using Visual Studio IDE

First we need to create the .sln files

Go to isomodel directory root (e.g. c:\git\isomodel\isomodel\) and make a binary directory

cd C:\git\idomodel\isomodel

mkdir bin

cd bin

Now you can cmake with a really long command line. You should copy the text below and paste into the command line window.

cmake ../src -G "Visual Studio 17 2022" -DBoost\_LIBRARY\_DIR="C:/local/boost\_1\_88\_0/lib64-msvc-14.3" -DBoost\_INCLUDE\_DIR="C:/local/boost\_1\_88\_0" -DGTEST\_ROOT="c:/local/googletest"

If things are working the output should look something like this:

Text

AI-generated content may be incorrect.

You might get warnings related to boost dependencies but that shouldn’t matter if the cmake file finishes without errors.

This process will have created a CMakeFiles directory and a bunch of other visual studio project files in the bin directory including “isomodel\_project.sln”. My directory looks like the following

Text

AI-generated content may be incorrect.

Load isomodel\_project.sln into Visual Studio and change build to Release, x64 and build the entire project

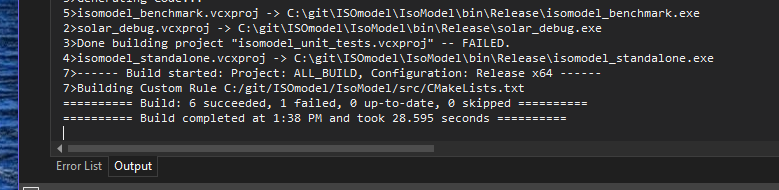
Graphical user interface, application

AI-generated content may be incorrect.

Graphical user interface, text, application

AI-generated content may be incorrect.

This will crunch for a while and give you many warnings and some errors, but hopefully you will get through it and the bottom of the output should look something like this:



The unit test project fails for now (still debugging that) but you should have 6 succeeded builds including isomodel\_standalone.exe as shown above.

## Testing the Build.

Now you need to copy some test data to the directory where the isomodel\_standalone.exe is located as the build scripts aren’t copying it (yet). In your terminal get to the isomodel\isomodel\bin directory and give the following commands

cd Release

cp ..\..\test\_data\SmallOffice\_v2.ism .

cp ..\..\test\_data\ORD.epw .

Now you are able to test the isomodel-standalone with the command

isomodel\_standalone.exe -h -i SmallOffice\_v2.ism

You should get something like the following out

