**Adding a Library**:

<https://youtu.be/abuCXC3t6eQ>

**Demo of VS-CMake-integration:**

[**https://youtu.be/OSvAeb99YcM?t=614**](https://youtu.be/OSvAeb99YcM?t=614)

**CMake Projects in VS**

<https://docs.microsoft.com/en-us/cpp/build/cmake-projects-in-visual-studio?view=vs-2019>

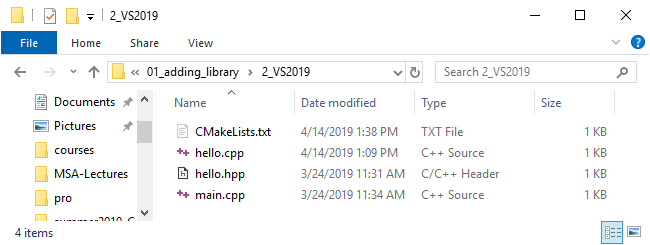
**VS-2019 CMake Project Settings UI**

<https://devblogs.microsoft.com/cppblog/introducing-the-new-cmake-project-settings-ui/>

**VS-Integration of CMake** (Project Setup)

<https://youtu.be/SYcTXK4q2Hg>

* **Local** directory-path :
  + D:\dev\build\2\_cmake\01\_adding\_library\2\_VS2019



* **CMakeLists.txt:**

cmake\_minimum\_required(VERSION 3.10.2)

project(MyProject VERSION 1.0.0)

add\_library(

    say-hello STATIC

    hello.hpp

    hello.cpp

)

# (<executable> <source>)

add\_executable(josh main.cpp)

# (<executable-linking-into> <link-interface-mode> <name-of-library-linking-into-executable>)

target\_link\_libraries(josh PRIVATE say-hello)

* **main.cpp**

#include "hello.hpp"

int main()

{

hello::say\_hello();

}

* **hello.cpp**

#include <iostream>

#include "hello.hpp"

void hello::say\_hello()

{

std::cout << "Hello\n";

}

* **hello.h**

#ifndef HELLO\_HPP\_INCLUDED

#define HELLO\_HPP\_INCLUDED

namespace hello

{

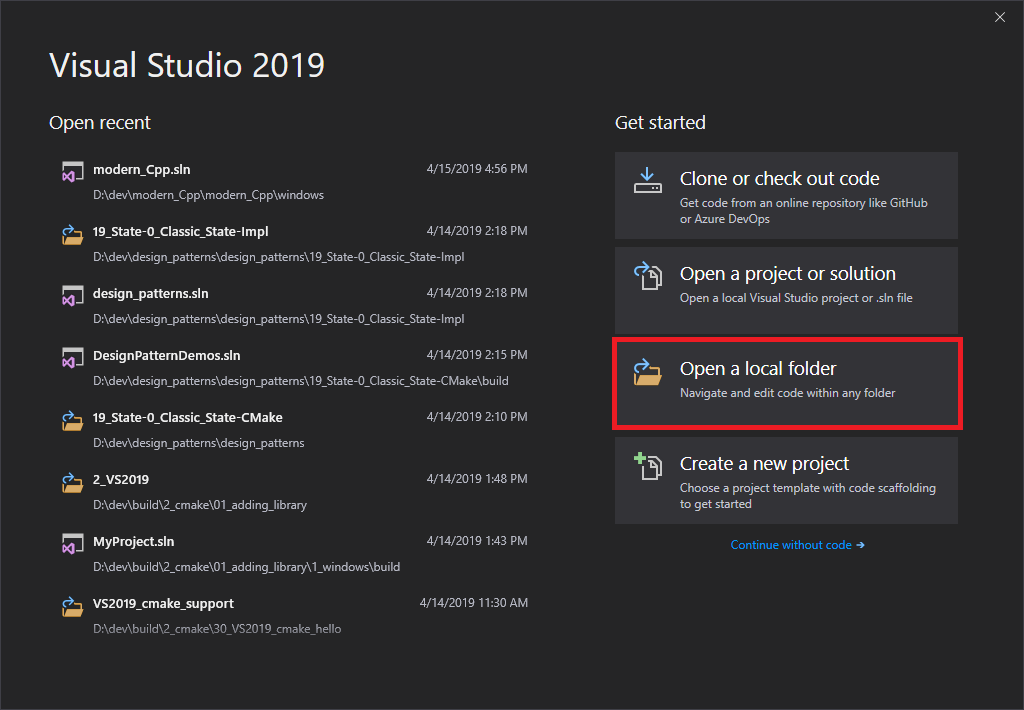
void say\_hello();

}

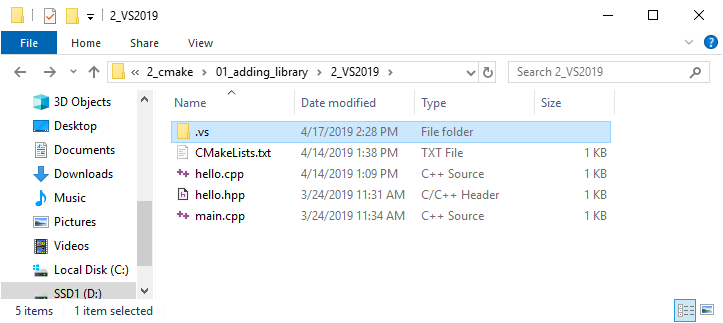
#endif

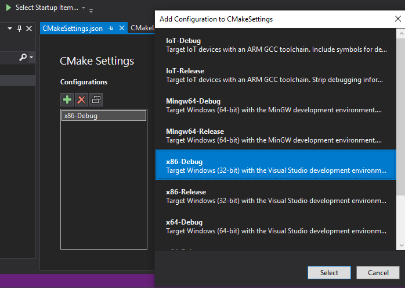
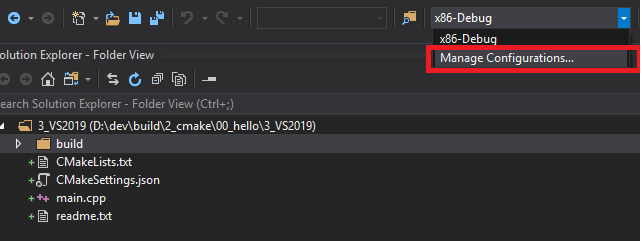
Open folder containing CMakeLists.txt

* Open folder containing the CMakeList.txt file.

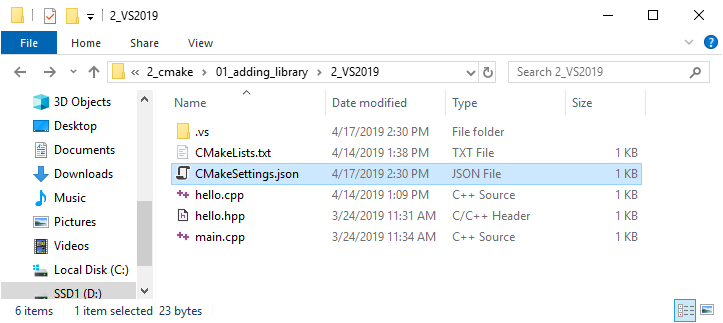


**Opening a directory containing the CMakeLists.txt file results in the .vs folder being created**:



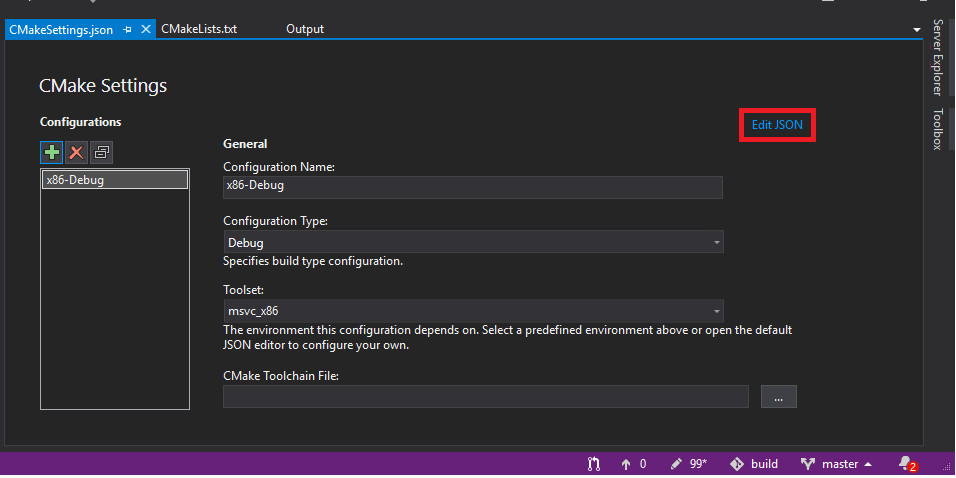
**Change to x86-debug: **

**By selecting “Manage Connections” the CMakeSettings.json file is created:**



**Edit the JSON file**:

[https://docs.microsoft.com/en-us/cpp/build/**configure-cmake-debugging-sessions**?view=vs-2019](https://docs.microsoft.com/en-us/cpp/build/configure-cmake-debugging-sessions?view=vs-2019)

****

Change the **buildRoot** and **installRoot**:

{

"configurations": [

{

"name": "x86-Debug",

"generator": "Ninja",

"configurationType": "Debug",

"buildRoot": **"${workspaceRoot}**\\build\\${name}",

"installRoot": "**${workspaceRoot}**\\install\\${name}",

"cmakeCommandArgs": "",

"buildCommandArgs": "-v",

"ctestCommandArgs": "",

"inheritEnvironments": [ "msvc\_x86" ],

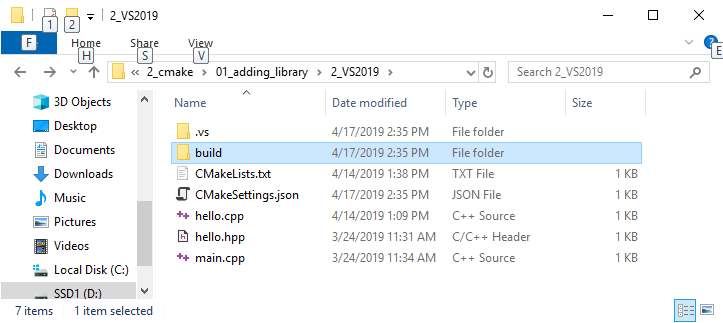
"variables": []

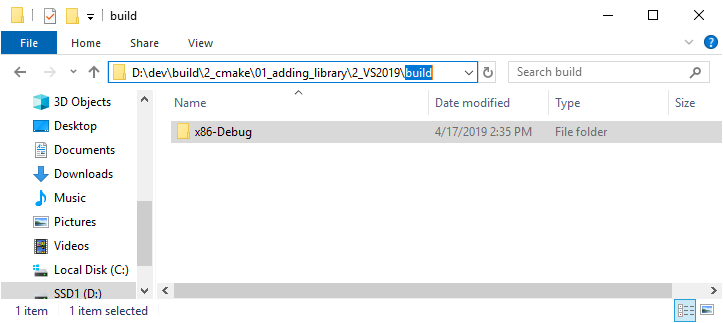
}

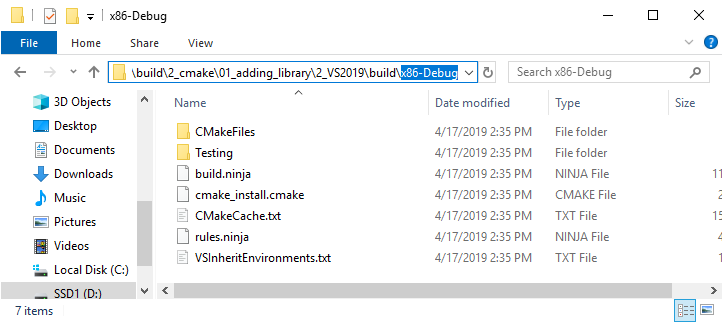
]

}

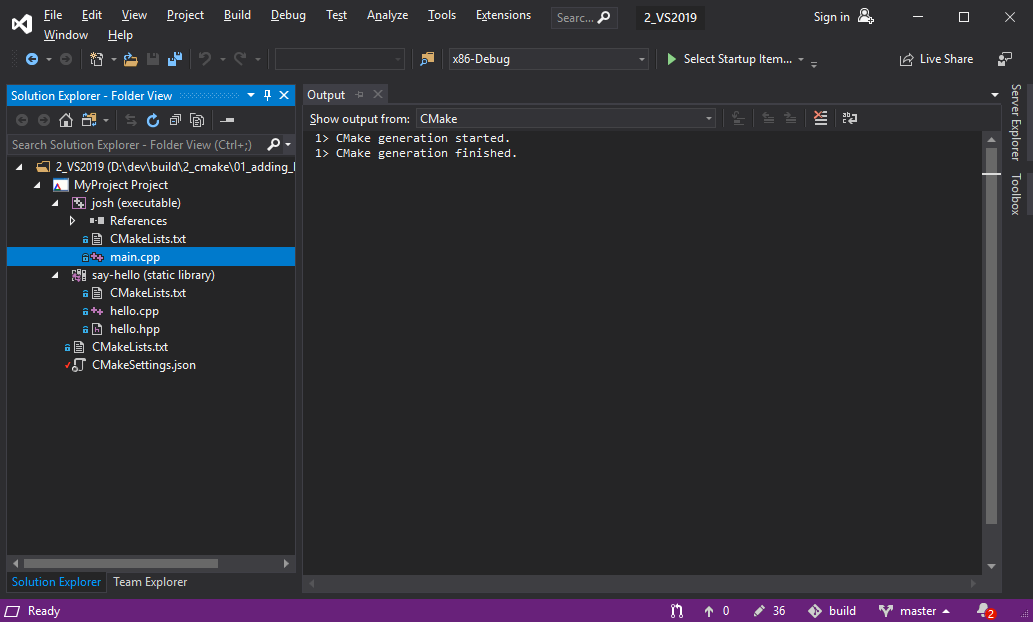
**By saving the edit to CMakeSettings.json the build directory is generated automatically** (before building):



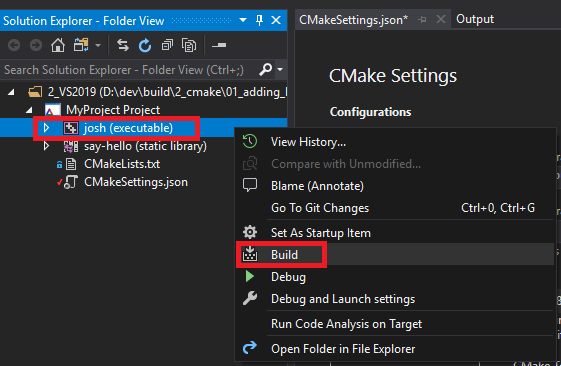




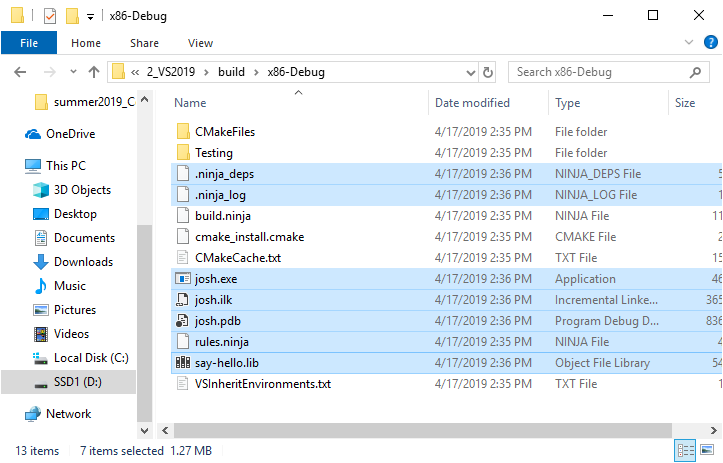
**CMakeView**:



Each **Target** can be built separately:

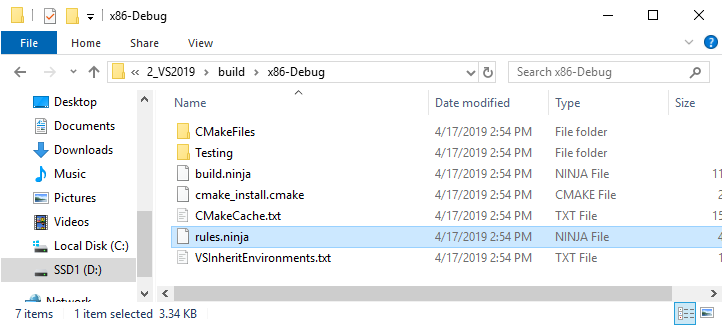


Building the executable target results in the following in the build directory:

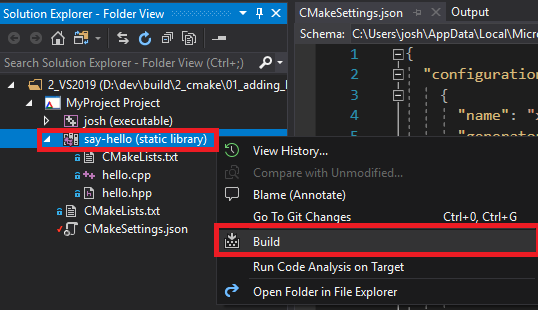


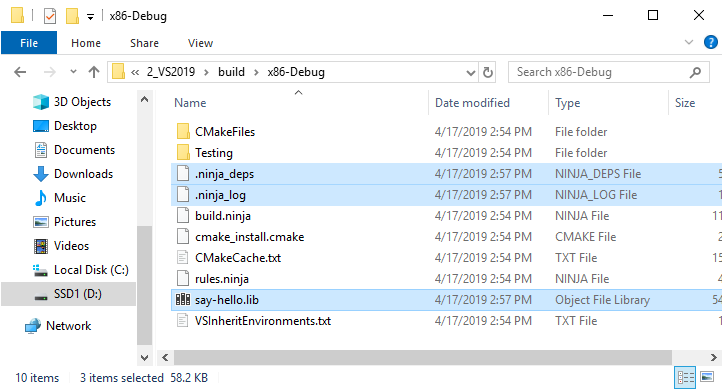
**NOTE:** rules.ninja was generated in previous step.

* Apparently, building the executable target automatically builds the static-lib target since we link to it in CMakeLists.txt.
* For completeness, the following step deletes all the generated files, and then first builds the static-lib target, followed by the executable target
  + To show a clear picture of the exact files generated from building the static-lib target
* Delete the built files (actually, stepped through all of the steps from the beginning):

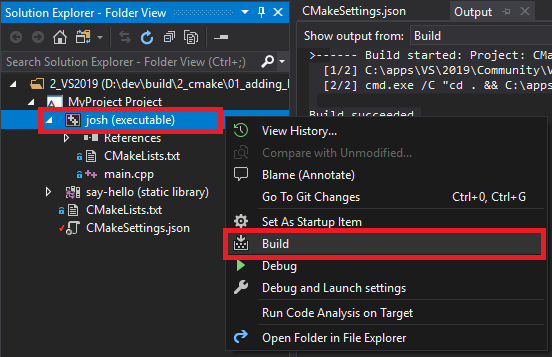


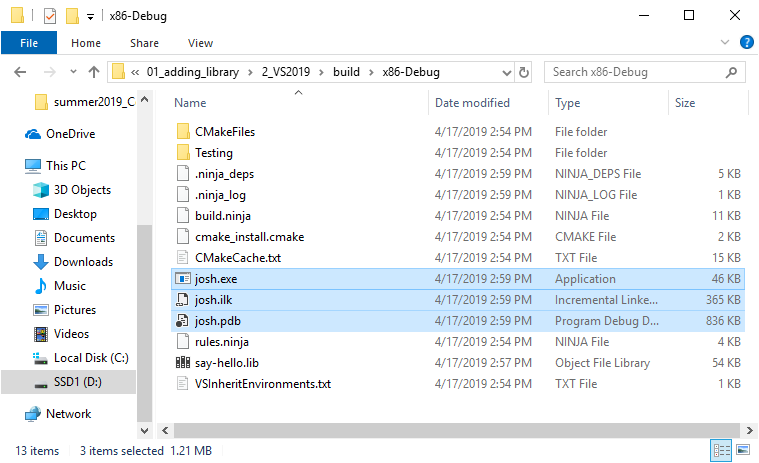
* Build the static-lib target





* Build the executable target





* Run the executable

