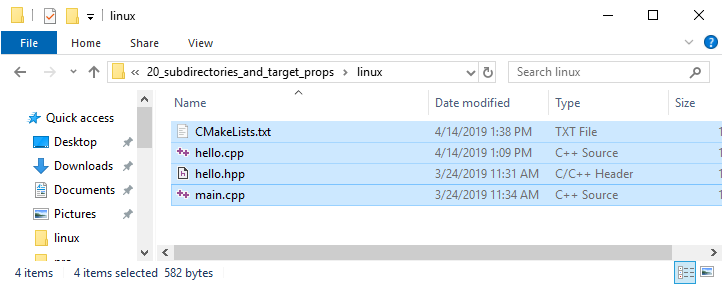
Subdirectories and Target Interface Properties

<https://youtu.be/SYgESCQeGJY>

**Linux:**

Start with project “adding a library”.



**hello.cpp**

#include <iostream>

#include "hello.hpp"

void hello::say\_hello()

{

    std::cout << "Hello\n";

}

**hello.hpp**

#ifndef HELLO\_HPP\_INCLUDED

#define HELLO\_HPP\_INCLUDED

namespace hello

{

    void say\_hello();

}

#endif

**main.cpp**

#include "hello.hpp"

int main()

{

    hello::say\_hello();

}

**CMakeLists.txt**

cmake\_minimum\_required(VERSION 3.10.2)

project(MyProject VERSION 1.0.0)

add\_library(

    say-hello STATIC

    hello.hpp

    hello.cpp

)

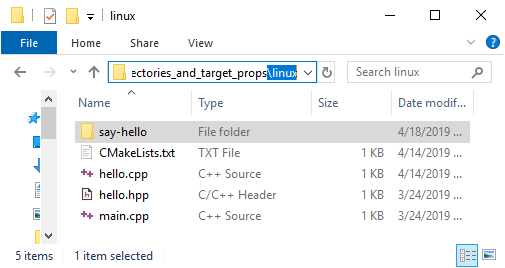
# (<executable> <source>)

add\_executable(josh main.cpp)

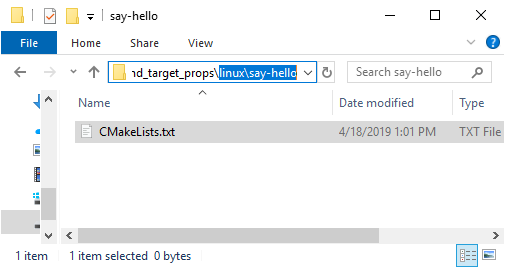
# (<executable-linking-into> <link-interface-mode> <name-of-library-linking-into-executable>)

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

**Create subdirectory named say-hello**



**Create CMakeLists.txt** **in new sub-directory**



**Modify root CMakeLists.txt:**

cmake\_minimum\_required(VERSION 3.10.2)

project(MyProject VERSION 1.0.0)

**# (<name-of-directory-relative-to-this-file>)**

**add\_subdirectory(say-hello)**

add\_library(

    say-hello STATIC

    hello.hpp

    hello.cpp

)

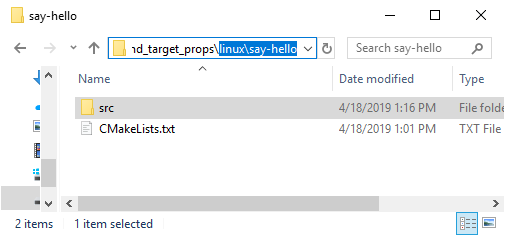
# (<executable> <source>)

add\_executable(josh main.cpp)

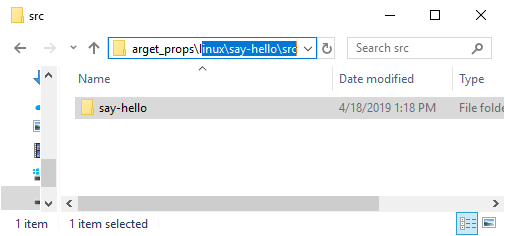
# (<executable-linking-into> <link-interface-mode> <name-of-library-linking-into-executable>)

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

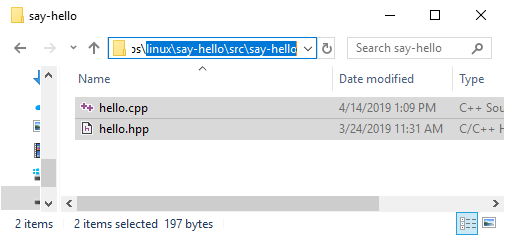
**Create sub-directory** linux\say-hello\**src**



**Create sub-directory** linux\say-hello\src\**say-hello**



**Move the .cpp and .hpp files corresponding to the library into this directory**



**Move add\_library** from **root** linux\**CMakeLists.txt** into **linux\say-hello\CMakeLists.txt**

* Change the paths of the files to be in relation to the root directory

linux\say-hello\**CMakeLists.txt**

add\_library(

    say-hello STATIC

**say-hello/src/**hello.hpp

**say-hello/src/**hello.cpp

)

**root** linux\**CMakeLists.txt** after mod:

cmake\_minimum\_required(VERSION 3.10.2)

project(MyProject VERSION 1.0.0)

# (<name-of-directory-relative-to-this-file>)

add\_subdirectory(say-hello)

#add\_library(

#   say-hello STATIC

#   hello.hpp

#   hello.cpp

#)

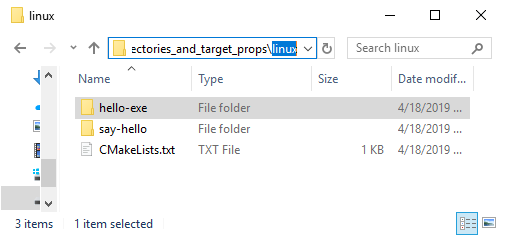
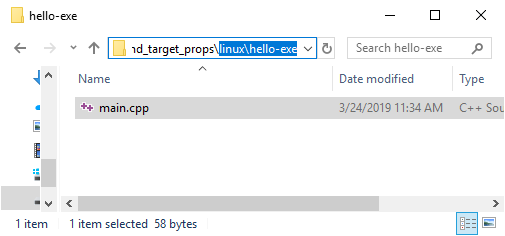
# (<executable> <source>)

add\_executable(josh main.cpp)

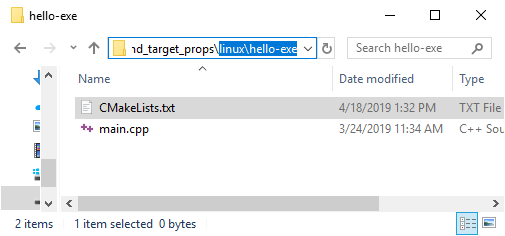
# (<executable-linking-into> <link-interface-mode> <name-of-library-linking-into-executable>)

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

**Move the main .cpp file into a new directory** linux\**hello-exe**:

**Add a CMakeLists.txt file to this directory**:



**Move add\_executable** from **root** linux\**CMakeLists.txt** into **linux\hello-exe\CMakeLists.txt**

* Change the paths of the files to be in relation to the root directory

linux\say-hello\**CMakeLists.txt**

# (<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)

**root** linux\**CMakeLists.txt** after mod:

cmake\_minimum\_required(VERSION 3.10.2)

project(MyProject VERSION 1.0.0)

# (<name-of-directory-relative-to-this-file>)

add\_subdirectory(say-hello)

## (<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)

**Add new add\_subdirectory** to **root** linux\**CMakeLists.txt** :

cmake\_minimum\_required(VERSION 3.10.2)

project(MyProject VERSION 1.0.0)

# (<name-of-directory-relative-to-this-file>)

add\_subdirectory(say-hello) # Needs to be first [see comment below]

**add\_subdirectory(hello-exe)**

NOTE:

Because the executable named **josh** (specified in the add\_executable line in linux\say-hello\**CMakeLists.txt**)links to the library **say-hello** (specified in the target\_link\_libraries line in the same file), the library target must be defined before CMake passes over the lines corresponding to the linux\say-hello\**CMakeLists.txt** file.

linux\say-hello\**CMakeLists.txt**

# (<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)**

**Define an include path for the library’s header file**

* Allows the #include "hello.hpp" to work
* Achieved by **creating a new** target\_link\_libraries **command to** linux\say-hello\**CMakeLists.txt**

linux\say-hello\**CMakeLists.txt**

add\_library(

    say-hello STATIC

    say-hello/src/hello.hpp

    say-hello/src/hello.cpp

)

# (<name-of-target> <interface-mode> <list-of-directories>)

target\_include\_directories(say-hello PUBLIC "${CMAKE\_CURRENT\_SOURCE\_DIR}/src")

#${CMAKE\_CURRENT\_SOURCE\_DIR} refers to the directory that owns the currently-being processed CMakeLists.txt file

# i.e., linux/say-hello

* Change the path in main .cpp file

linux\hello-exe\main.cpp

#include "**say-hello/**hello.hpp"

int main()

{

    hello::say\_hello();

}

* Create directory named **build** in root

josh:/mnt/d/dev/build/2\_cmake/20\_subdirectories\_and\_target\_props/linux**$ mkdir build**

* cd into build directory and run CMake on the root CMakeLists.txt file

josh:/mnt/d/dev/build/2\_cmake/20\_subdirectories\_and\_target\_props/linux/build**$ cmake ..**

Issue in the final ste[

target\_include\_directories(say-hello PUBLIC "${CMAKE\_CURRENT\_SOURCE\_DIR}/src")

**Directory-Structure**