2021年10月10日 18:14

Vim

2021年10月10日 18:14

vim命令
i进入插入模式
esc进入命令模式
输入： wq
保存退出。

cmake管理工程

2021年10月10日　　18:27

1、touch创建CMakeLists.txt文件

```
liu@liu:~/shenlan$ touch CMakeLists.txt
liu@liu:~/shenlan$ vim CMakeLists.txt
liu@liu:~/shenlan$ cmake .
-- The C compiler identification is GNU 7.5.0
-- The CXX compiler identification is GNU 7.5.0
-- Check for working C compiler: /usr/bin/cc
-- Check for working C compiler: /usr/bin/cc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Detecting C compile features
-- Detecting C compile features - done
-- Check for working CXX compiler: /usr/bin/c++
-- Check for working CXX compiler: /usr/bin/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Configuring done
-- Generating done
-- Build files have been written to: /home/liu/shenlan
```

2、vim编辑CMakeLists.txt

3、

```
project(helloSLAM)

add_executable(sayHello main.cpp)
```

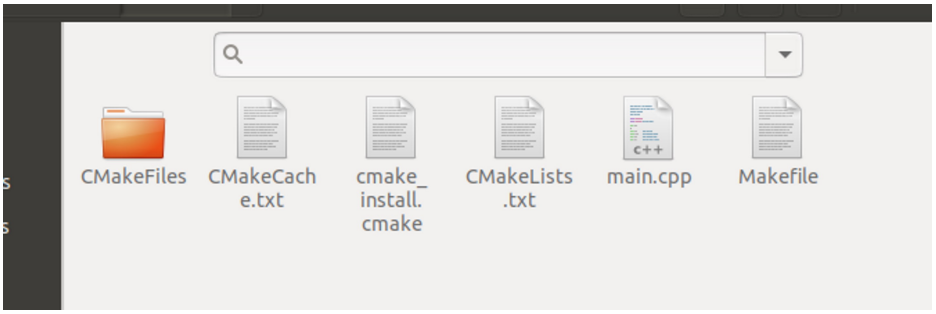
projec（）声明一个cmake工程

#添加一个可执行程序

#语法：add_executable(程序名 源代码文件)

4、在当前目录下 调用 cmake .

生成makefile文件：



5、调用make命令对工程进行编译

```
liu@liu:~/shenlan$ make
Scanning dependencies of target sayHello
[ 50%] Building CXX object CMakeFiles/sayHello.dir/main.cpp.o
[100%] Linking CXX executable sayHello
[100%] Built target sayHello
liu@liu:~/shenlan$
```

6、./sayHello执行生成程序。

上述方式cmake生成的中间文件还留在代码文件中，一种更好的方式是让这些中间文件放在一个中间目录中，在编译成功后将其删除即可。

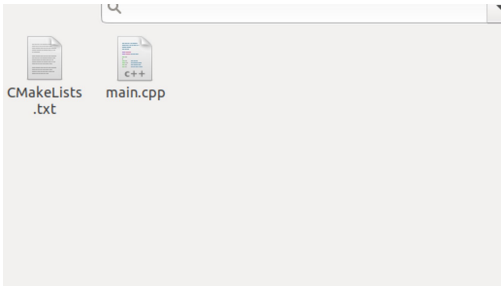
1.mkdir build

2.cd build

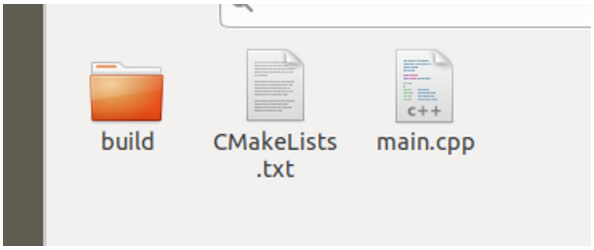
3.cmake ..

4.make

1、mkdir build

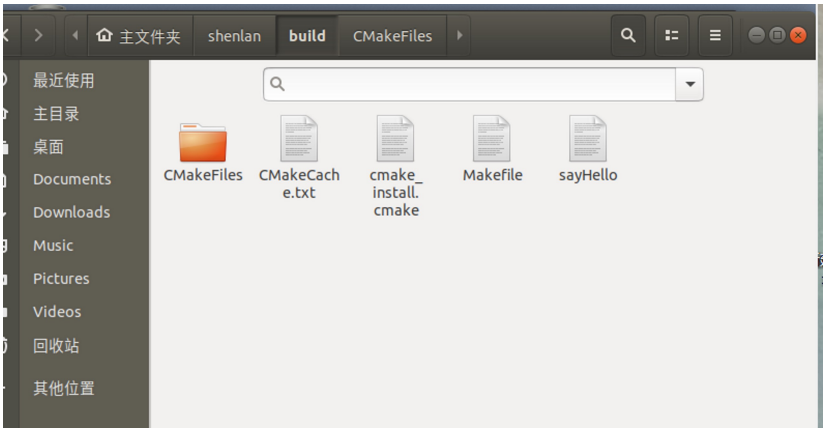


2、cd build



3、 cmake ..

4、make



build目录下生出可执行文件sayHello

2.4.4使用库

2021年10月10日 19:18

```
liu@liu:~/shenlan$ mkdir include src
liu@liu:~/shenlan$ touch include/Hello.h src/Hello.cpp
liu@liu:~/shenlan$ vim include/Hello.h
liu@liu:~/shenlan$ vim src/Hello.cpp
liu@liu:~/shenlan$ vim main.cpp
```

1、include目录下创建头文件Hello.h

```
#pragma once

void printHello();
```

2、src目录下创建库文件 Hello.cpp

```
#include<iostream>
using namespace std;
void printHello()
{
        cout<<"hello"<<endl;
}
```

3、main函数使用库函数 main.cpp

```
#include "Hello.h"

int main()
{
        printHello();


        return 0;
}
```

4、CMakeLists.txt

```
project(helloSLAM)

add_library(libHello src/Hello.cpp)

add_executable(useHello main.cpp)
target_link_libraries(useHello libHello)
```

```
project(helloSLAM)
include_directories("include")

add_library(libHello src/Hello.cpp)

add_executable(useHello main.cpp)
target_link_libraries(useHello libHello)
~
~
~
```

add_library将Hello.cpp编译成一个叫做”libHello“的库 后面的liblibHello.a文件

add_executable 将源代码文件main.cpp编译成可执行程序 useHello

target_link_libraries 将useHello程序可以顺利使用libHello库中的代码。

注意：要将incldue目录加入引用文件中。第二句

cd build

cmake ..

make

ll

```
liu@liu:~/shenlan/build$ ll
总用量 72
drwxrwxr-x 3 liu liu  4096 10月 10 20:01 ./
drwxrwxr-x 5 liu liu  4096 10月 10 20:01 ../
-rw-rw-r-- 1 liu liu 12937 10月 10 20:01 CMakeCache.txt
drwxrwxr-x 7 liu liu  4096 10月 10 20:01 CMakeFiles/
-rw-rw-r-- 1 liu liu  1486 10月 10 19:14 cmake_install.cmake
-rw-rw-r-- 1 liu liu  2936 10月 10 20:01 liblibHello.a
-rw-rw-r-- 1 liu liu  5531 10月 10 20:01 Makefile
-rwxrwxr-x 1 liu liu  8920 10月 10 19:15 sayHello*
-rwxrwxr-x 1 liu liu 13096 10月 10 20:01 useHello*
```

```
liu@liu:~/shenlan/build$ ./useHello
hello
```