# 跨系统编程说明

## 工程管理脚本CmakeList

Cmakelist.txt为调用codeit库、管理用户工程的脚本，可以实现跨平台工程管理，在每一个工程中都需要添加。内容如下：（详见于/跨系统编程示例 /CMakeList.txt）

# cmake版本要求：3.12及3.12以上

cmake\_minimum\_required(VERSION 3.12)

# 工程命名为“test”，如果用户根据实际项目情况重新命名，需要对本cmakelists中的“test”重新命名。"test"分别位于project、add\_executable、target\_link\_libraries三个函数中

project(test)

# codeit use c++ 17 standard——采用C++17标准

set(CMAKE\_CXX\_STANDARD 17)

#set函数将工程中的.cpp和.h文件的全部径添加到SOURCE\_FILES，方便程序引用(建议用户在src文件夹下创建.cpp和.h文件，方便管理)

set(SOURCE\_FILES src/main.cpp)

add\_executable(test ${SOURCE\_FILES})

if(UNIX)

target\_link\_libraries(test ${codeit\_LIBRARIES} atomic)

endif(UNIX)

IF(WIN32)

target\_link\_libraries(test ${codeit\_LIBRARIES})

endif(WIN32)

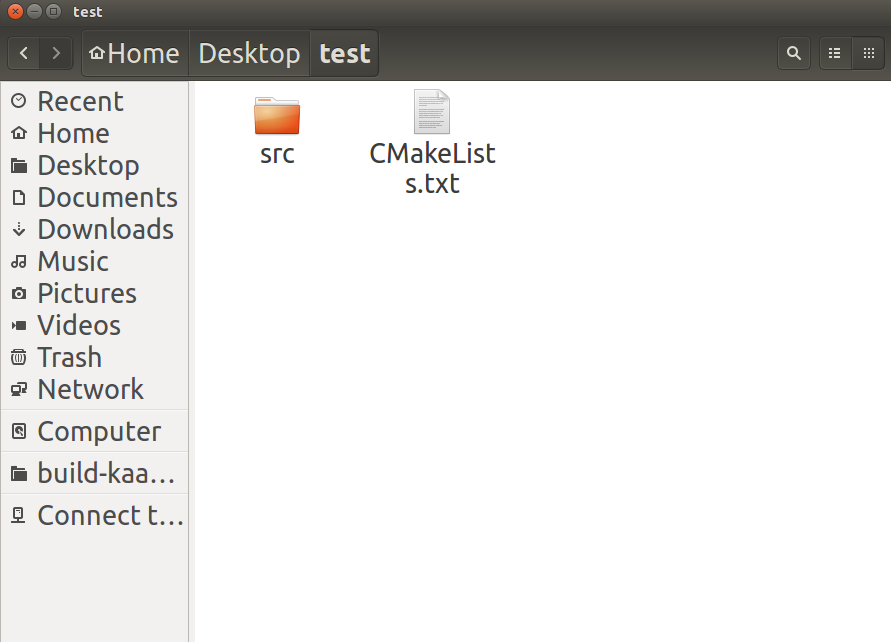
## 跨系统创建用户工程

### linux下创建工程

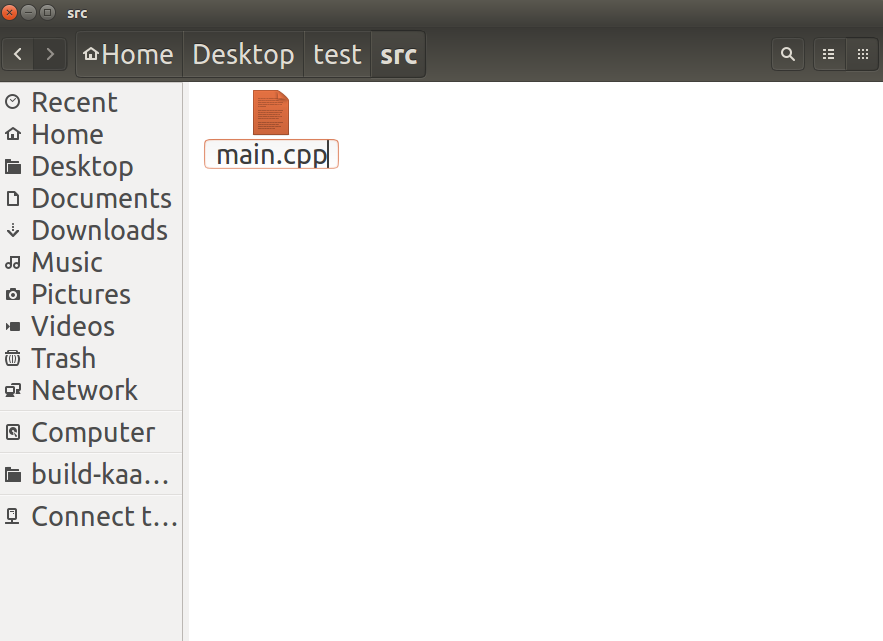
（1）在桌面(或者用户指定的路径)创建test工程文件夹



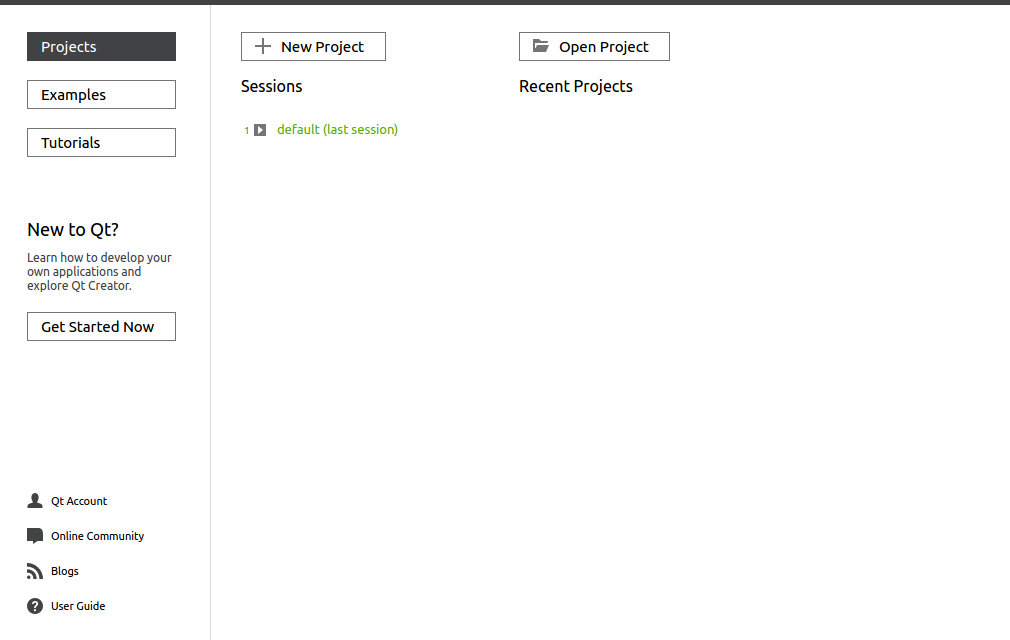
（2）在test工程文件夹下放置自带的cmakelist.txt；然后创建src文件夹，用于放置用户创建的工程文件。



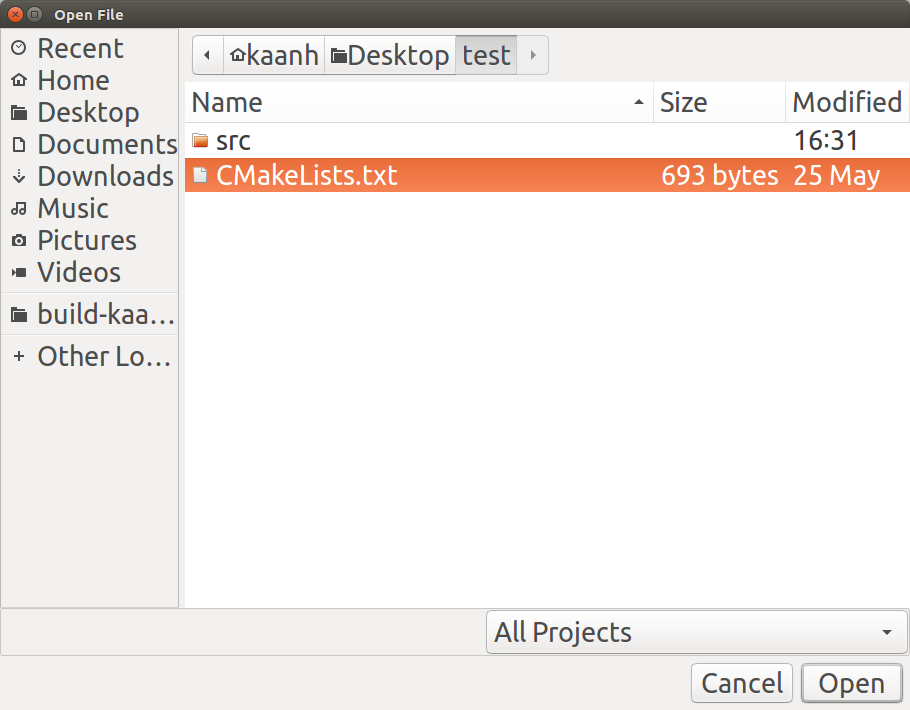
（3）在/test/src文件夹下，创建一个空文件,命名为main.cpp。



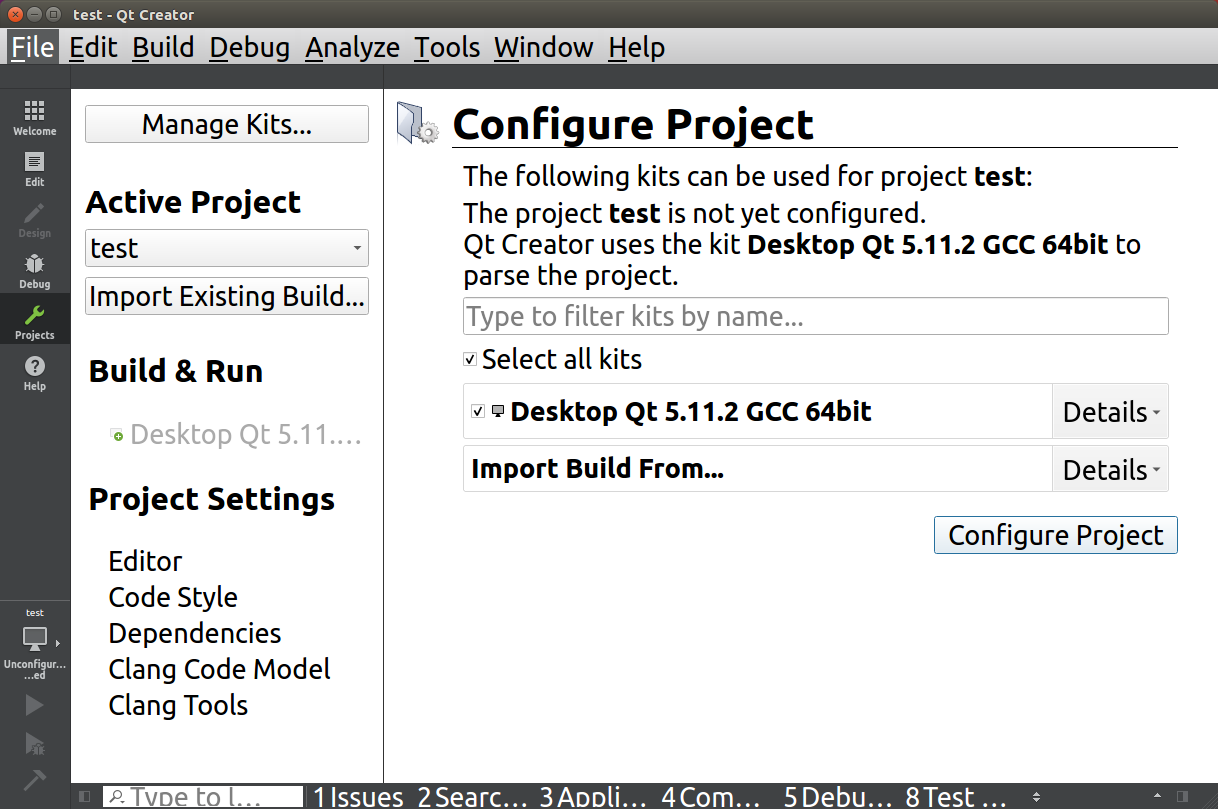
（4）通过桌面左侧任务栏快捷图标"Qt"，或者在安装路径下（默认安装路径：/opt/Qt5.11.2/Tools/QtCreator/bin/qtcreator），双击运行qt creator，然后选择“open project”。



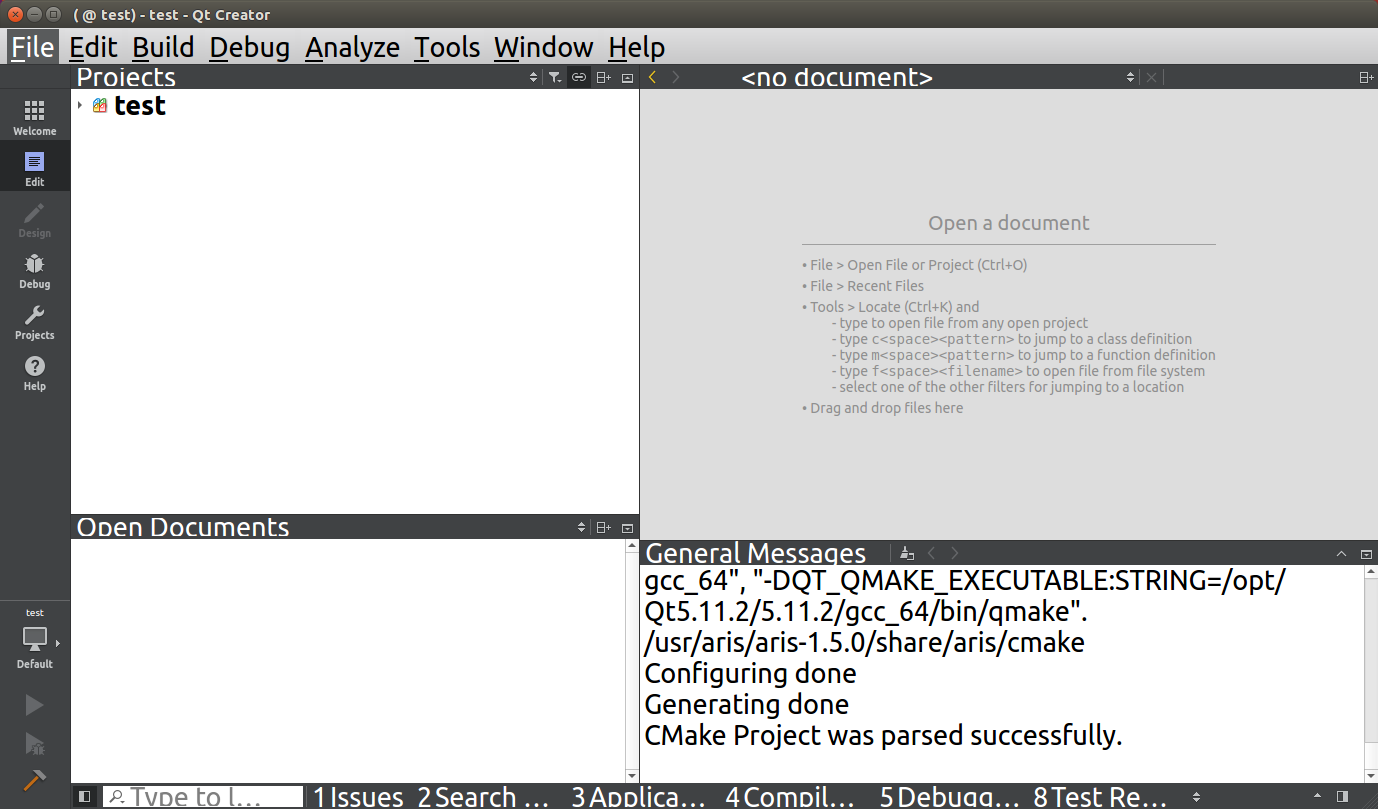
（5）定位到/Desktop/test文件夹下，选中CMakeList.txt文件夹，最后点击“Open”。



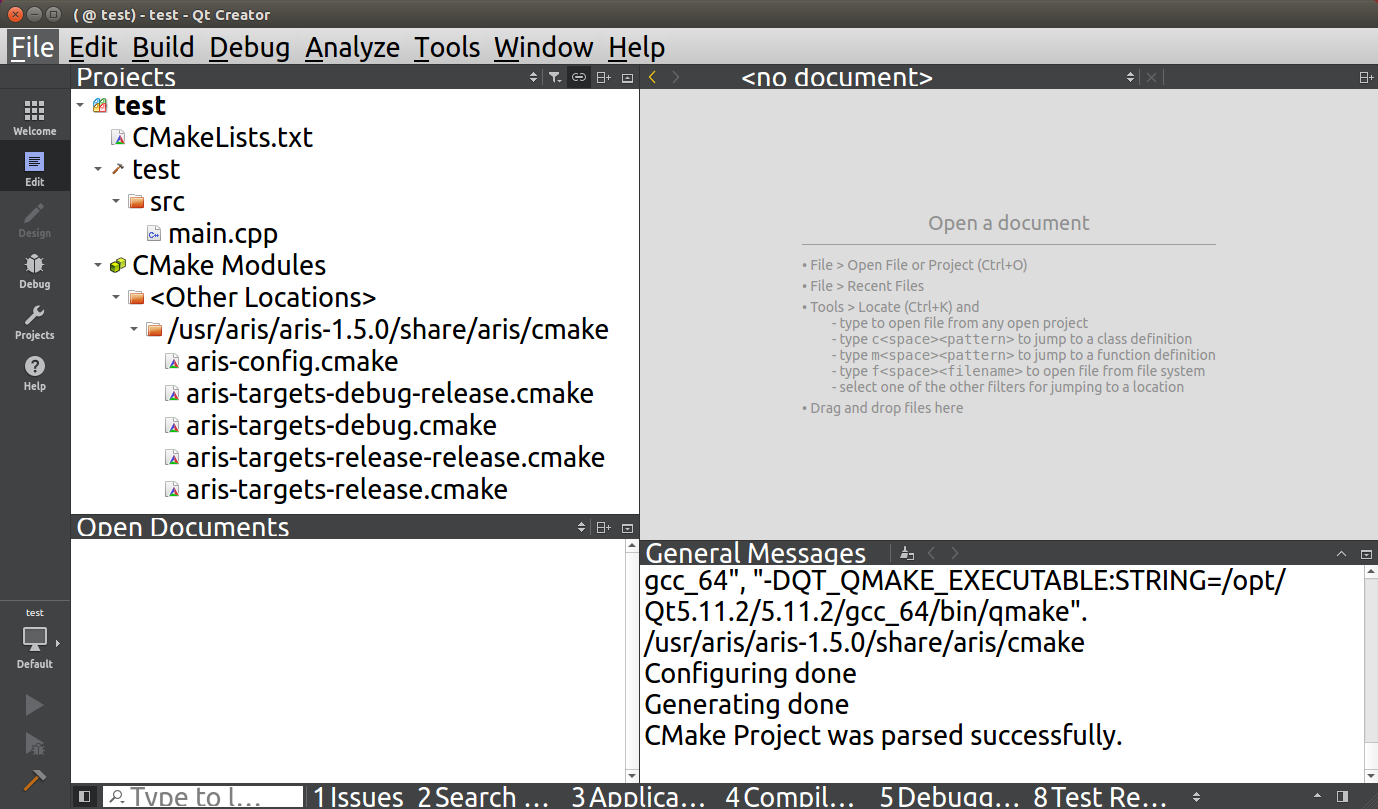
(6)点击“Configure Project”配置工程



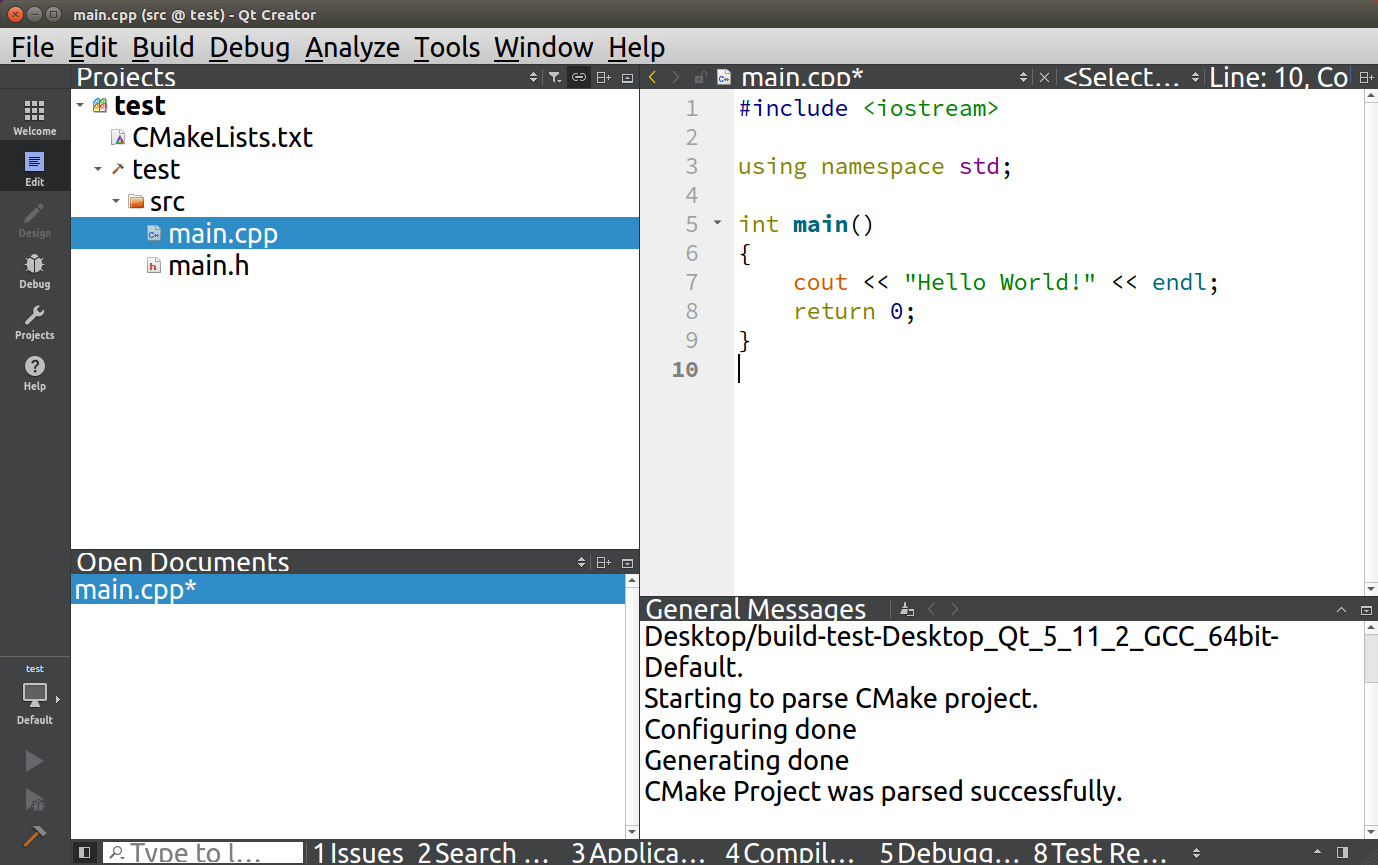
（7）配置后，工程会自动运行CMakeList脚本，或者通过菜单Build——run cmake运行CMakeList脚本



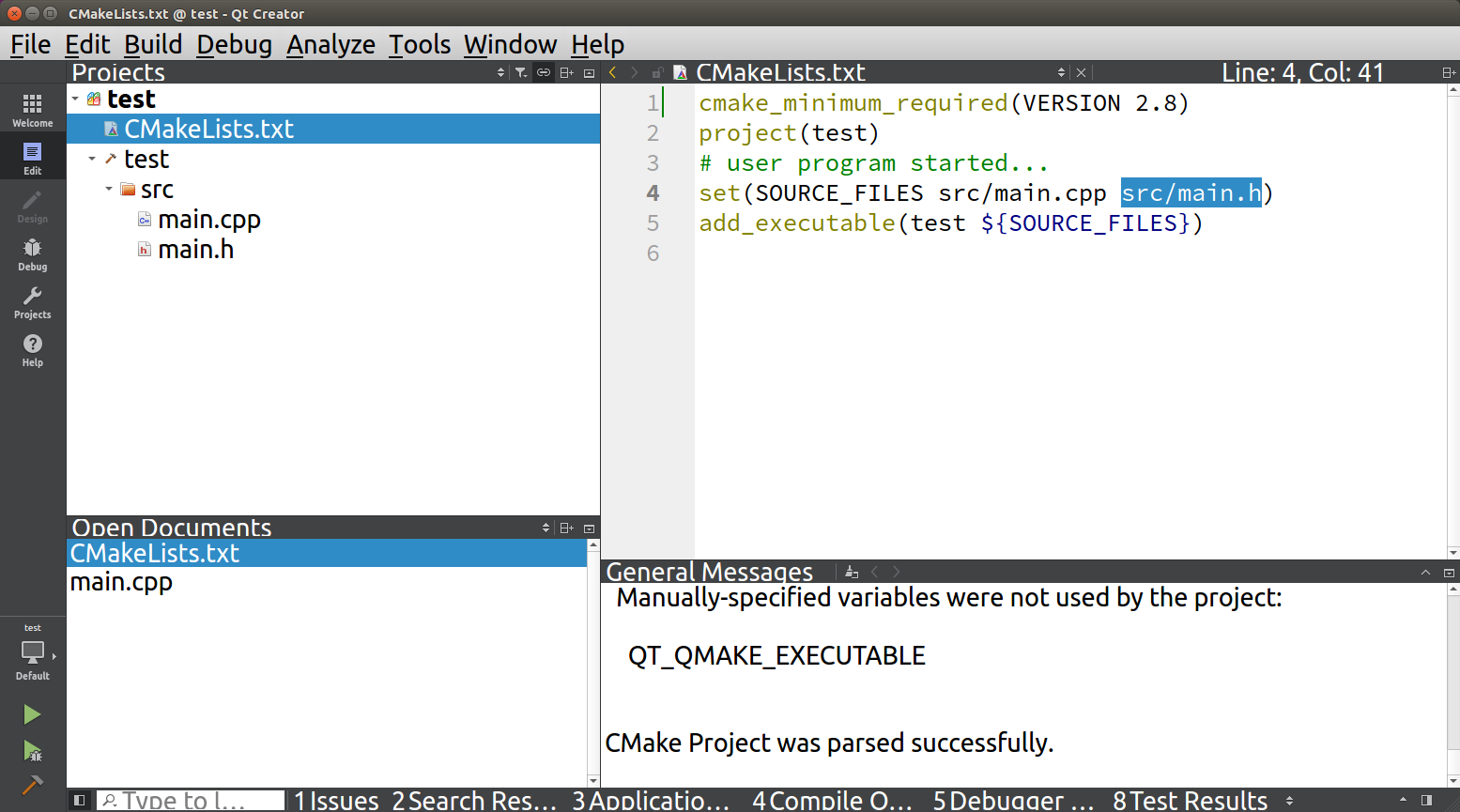
（8）步骤（6）的脚本运行后，工程test会索引到关联的工程文件，具体如下



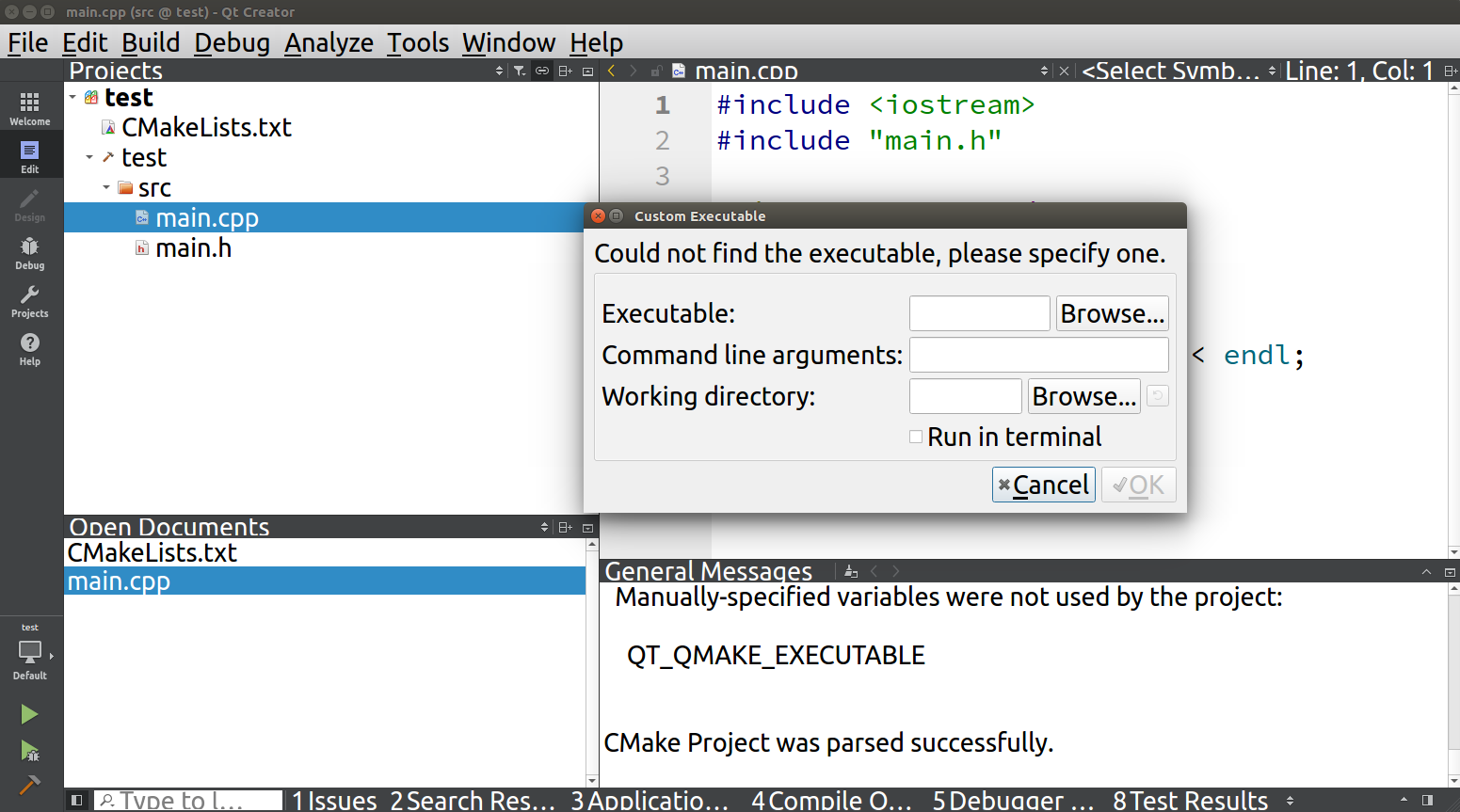
（9）在Qt creator界面中双击main.cpp文件，编写用户程序，本例是简单的hello world---main.cpp文件.



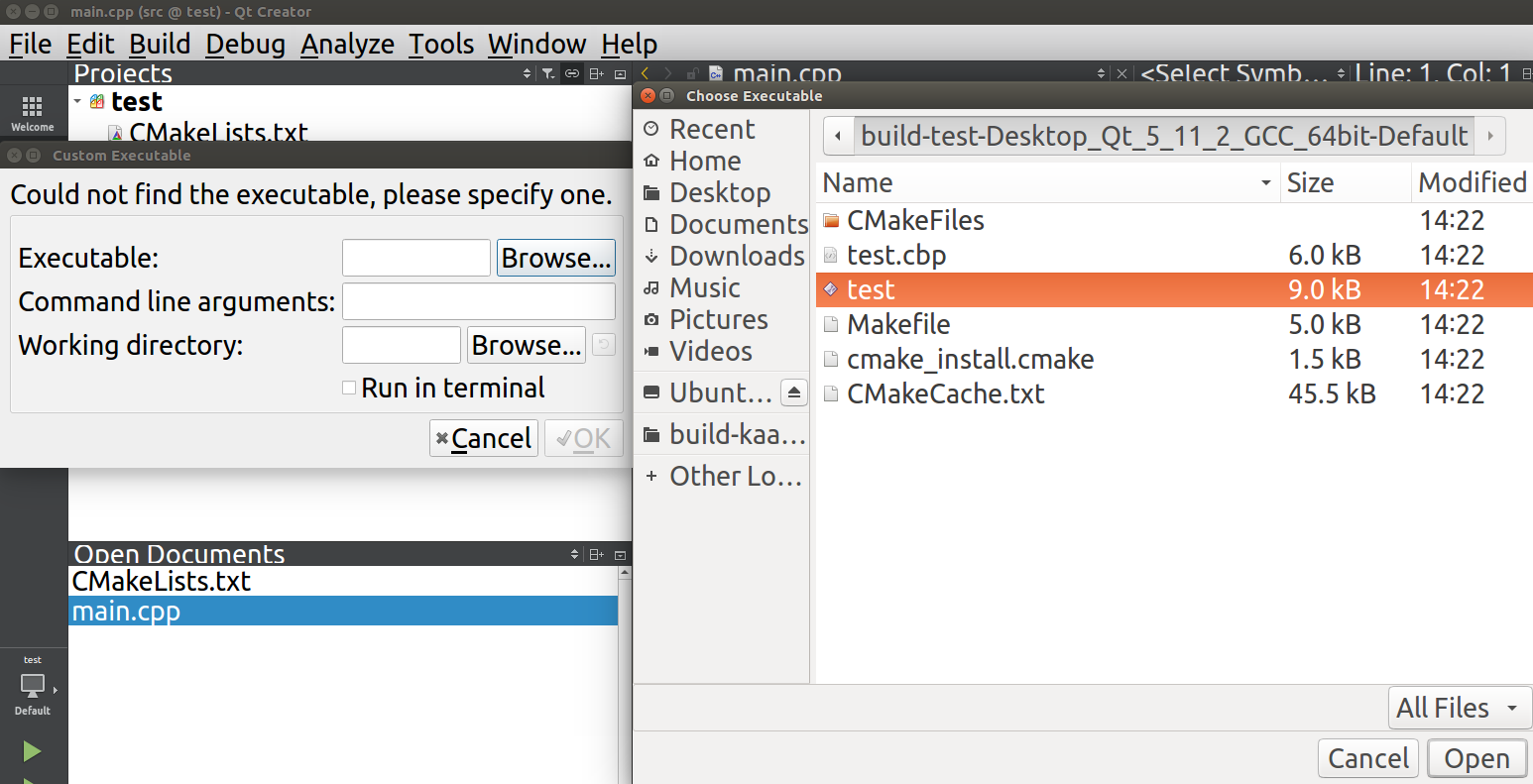
（10）用户也可以在src文件路径下新建.cpp或者.h文件，本例创建一个空头文件main.h。同时，用户需要更新Cmakelist.txt文件内容——添加src/main.h到SOURCE\_FILES中。最后，点击左下角标红按钮运行程序



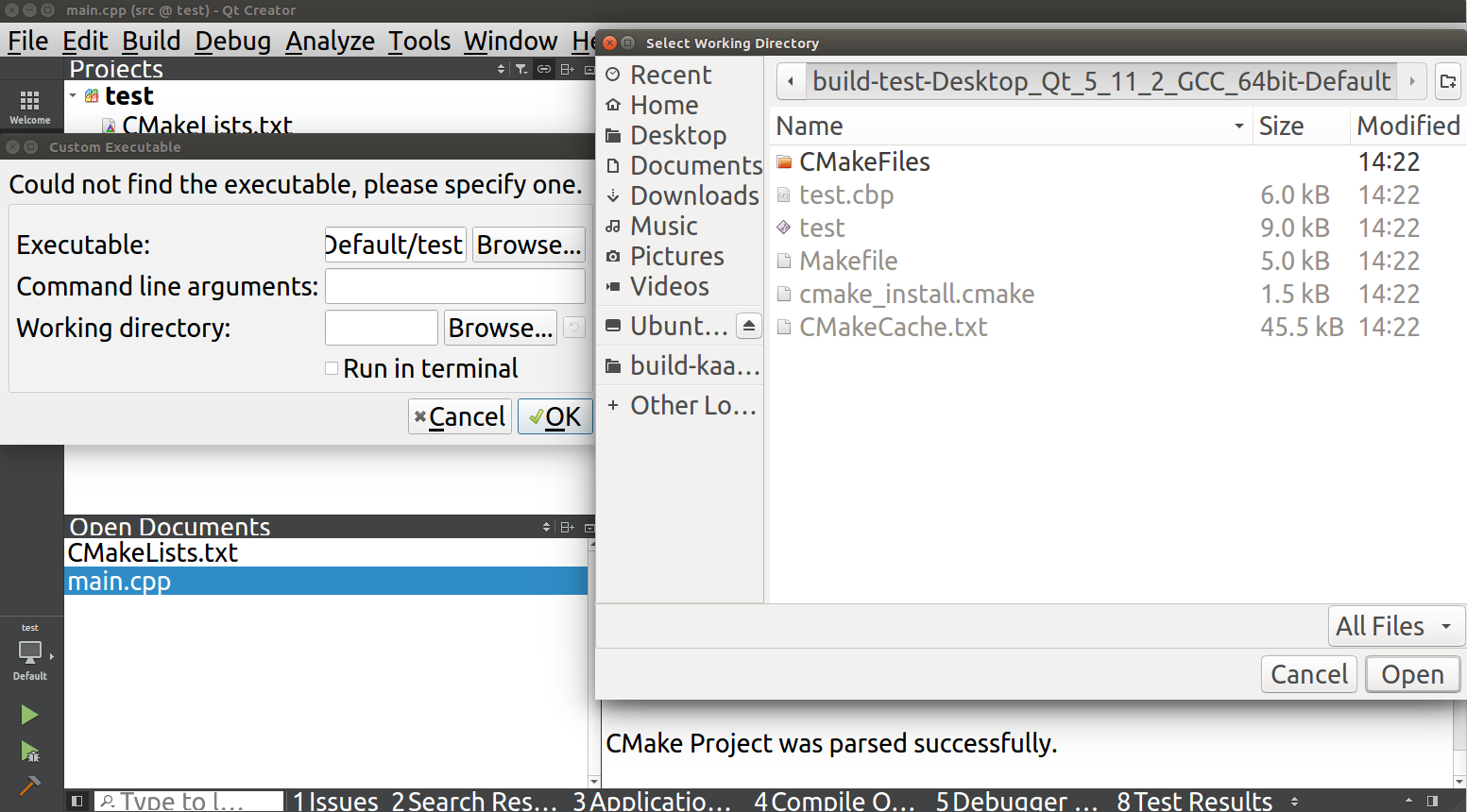
（11）如果弹出如下界面，说明对应的exe、工作路径没有指定，请执行后续步骤。否则，请忽略后续步骤。



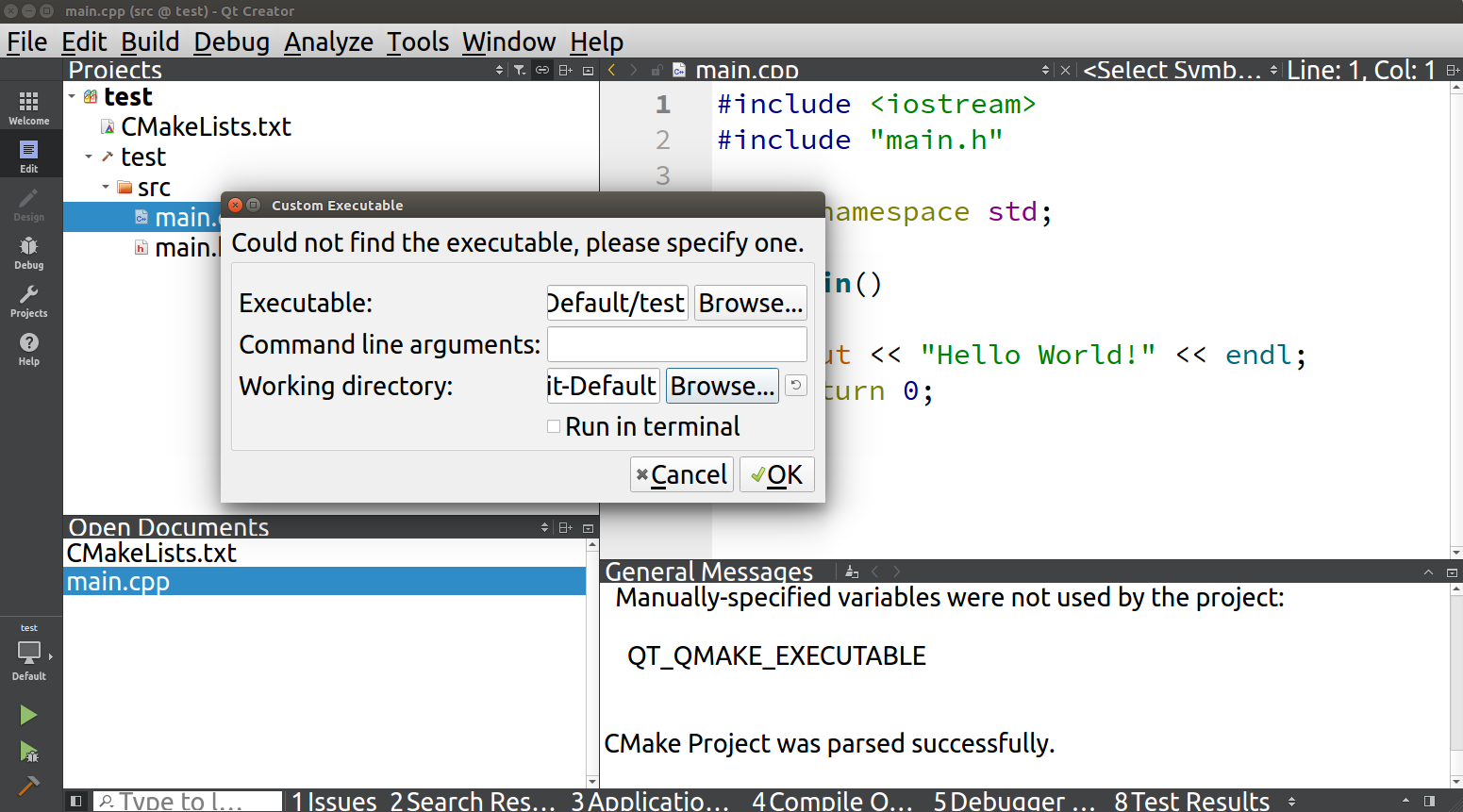
（12）Executable项，索引到程序的编译结果文件夹路径“/Desktop/build-test-Desktop-Default/test”



（13）Working directory项，索引到程序的编译结果文件夹路径“/Desktop/build-test-Desktop-Default”



（14）点击“OK”，即可运行程序



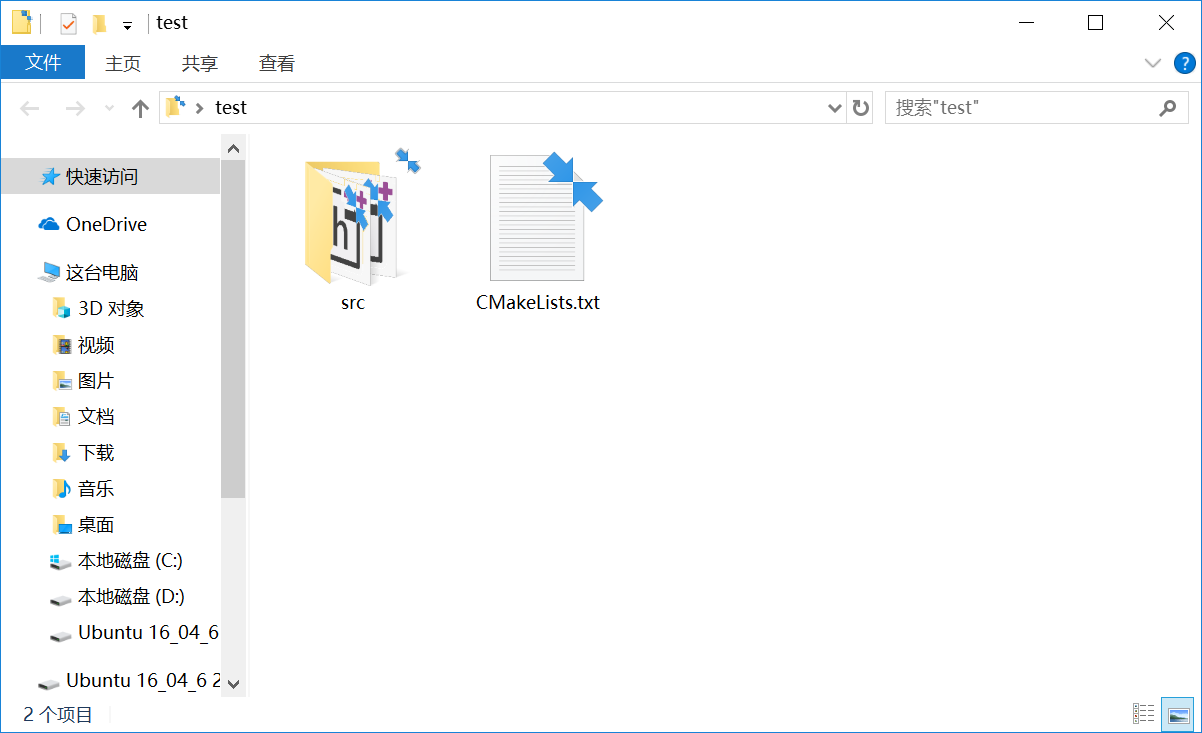
### Windows下创建工程

（1）在桌面(或者用户指定的路径，注意路径不要包含中文和空格)创建test程

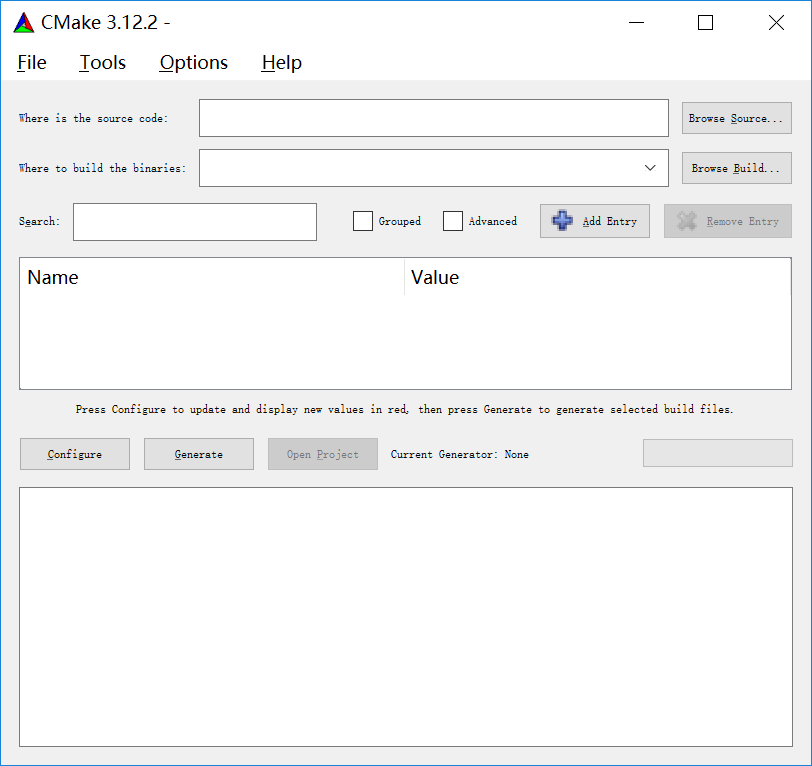
文件夹以及test\_build文件夹。



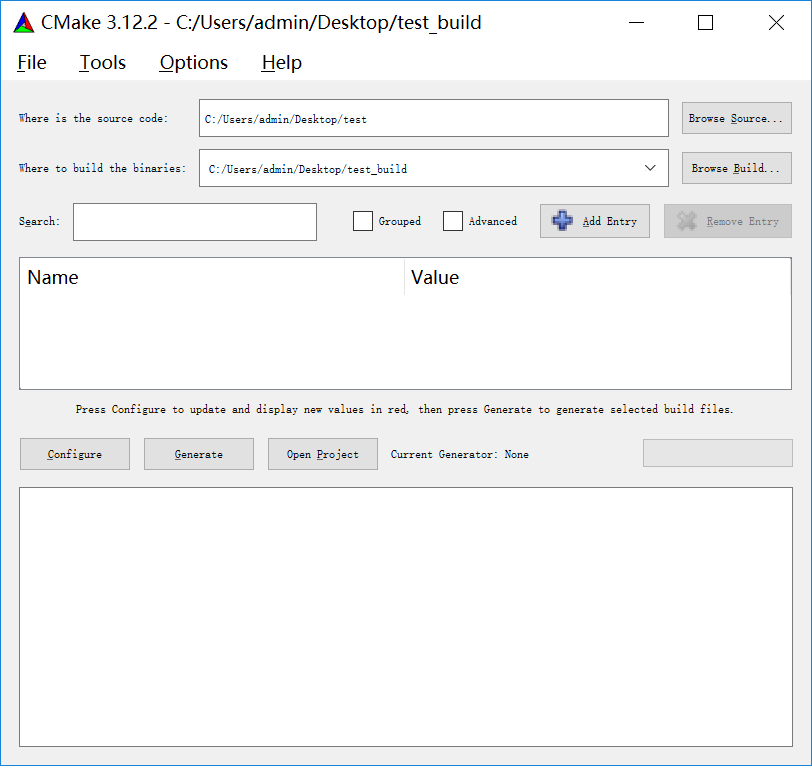
（2）在test工程文件夹下放置自带的cmakelist.txt；然后创建src文件夹，用于放置用户创建的工程文件(.cpp文件和.h/.hpp文件)，本例中src文件夹下放置自带的工程文件，路径为 /test/src (main.cpp)。



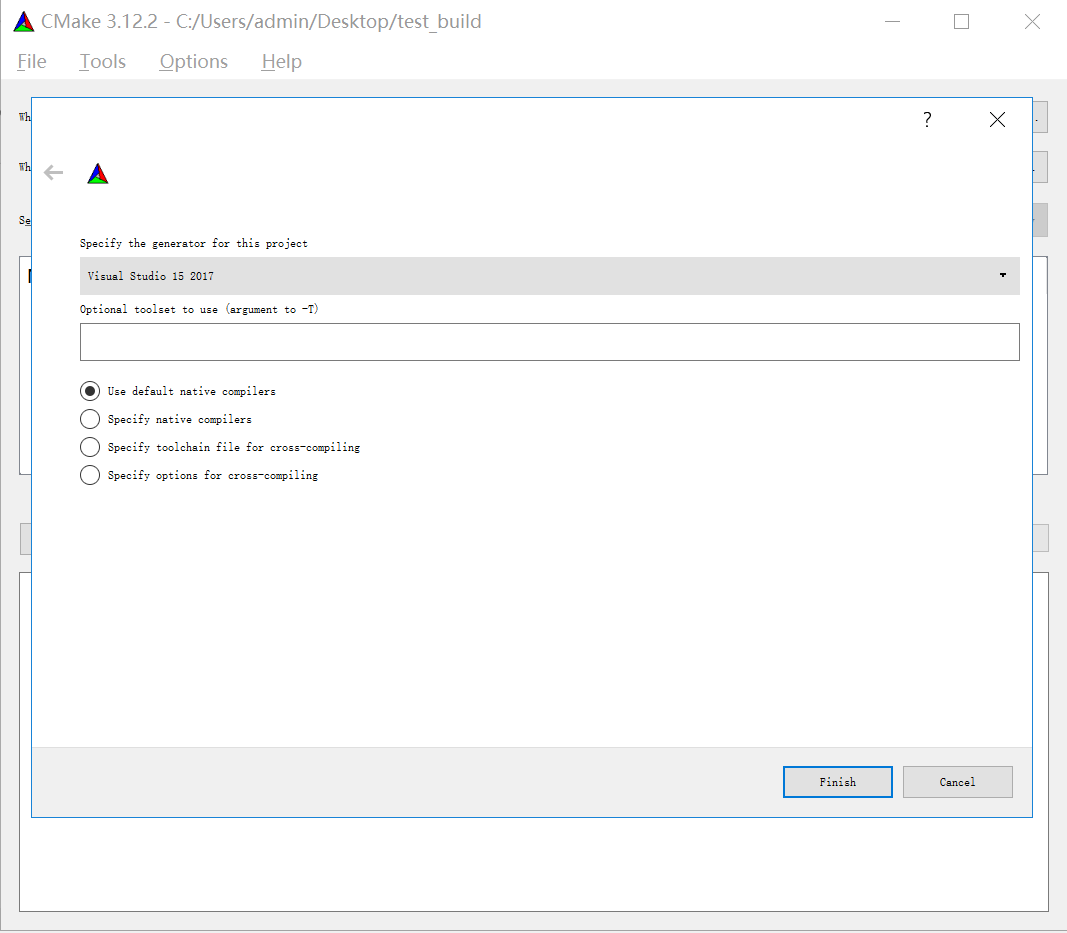
（3）在example/cmake-3.12.2-win64-x64/cmake-3.12.2-win64-x64/bin目录下，双击cmake-gui.exe，打开cmake界面。



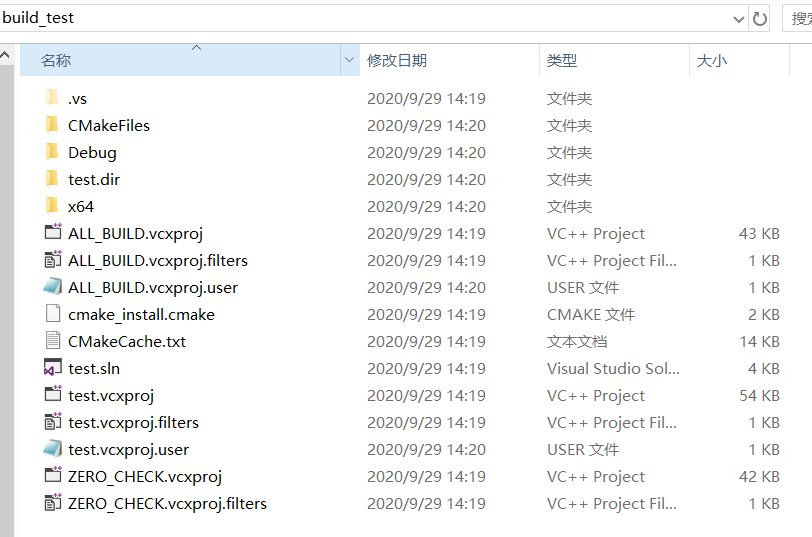
（4）在第一行填入test文件目录，在第二行填入test\_build文件目录。



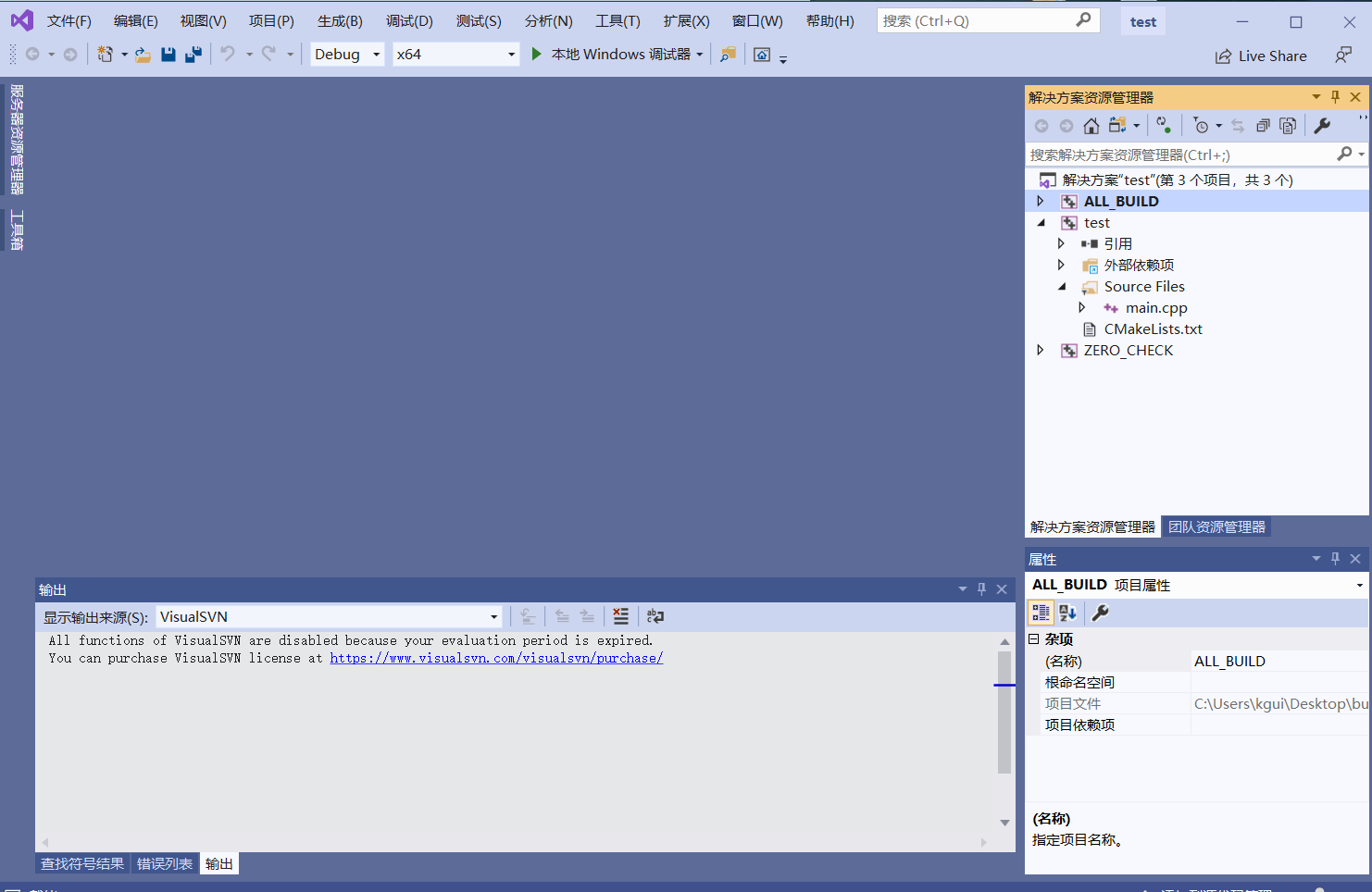
（5）单击configure配置工程,弹出如下图窗口，第一个选项下拉符号里选择visual studio 15 2017（选2017及以上版本），第二个选项选择第一个use default native compilers,然后单机finish,系统会自动运行CmakeList脚本。



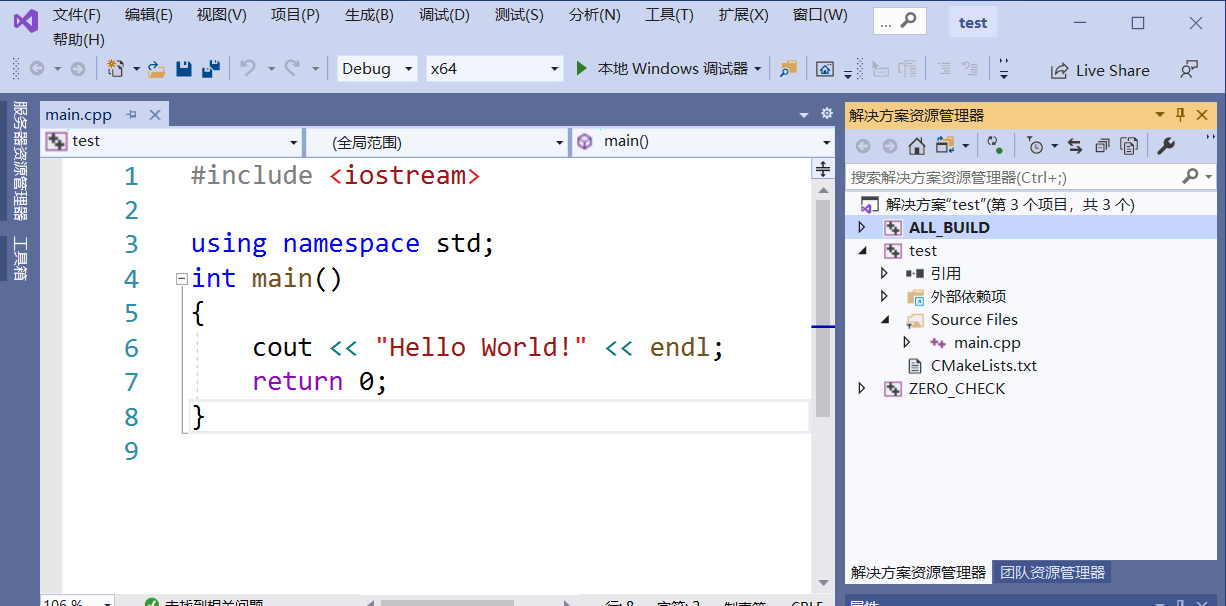
（6）单击generate后，build\_test文件夹下会生成一下文件：



（7）然后单击CMake软件界面上的open project打开工程文件或者在test\_build文件夹下双击test.sln打开工程文件（.sln是vs工程文件）.步骤（6）的脚本运行后，工程 test 会索引到关联的工程文件，具体如下。



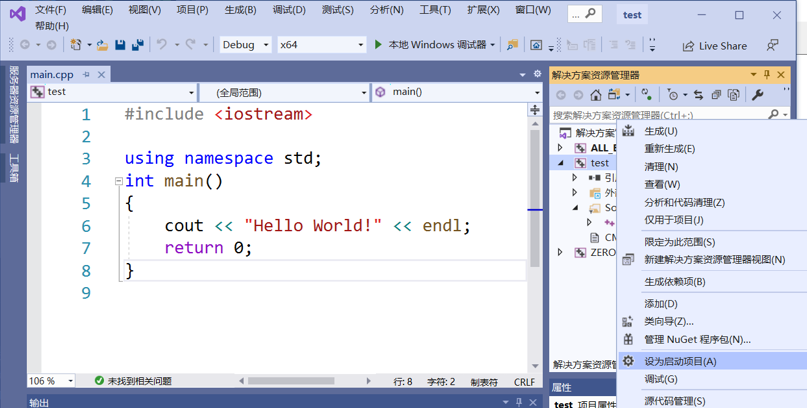
（8）在 vs2017 界面中双击打开 main.cpp 文件，编写用户程序。



（9）用户也可以在 src 文件路径下新建.cpp 或者.h 文件，例如新建main.h。同时，用户需要更新 Cmakelists.txt 文件相关内容——添加

src/main.h 到 SOURCE\_FILES 中，再单击上方菜单栏生成--生成解决方案（重新build CmakeLists.txt）。

（10）右键单击用户需要的工程，然后点击设为启动项目。设成功后，test字体会加粗。



（11）回到vs,然后点击上方绿色按钮（本地windows调试器）运行程序。