Clion Usage Guide

This guide tell you about basic usage of Clion.

If you cannot see the pictures clearly, try to goto this page

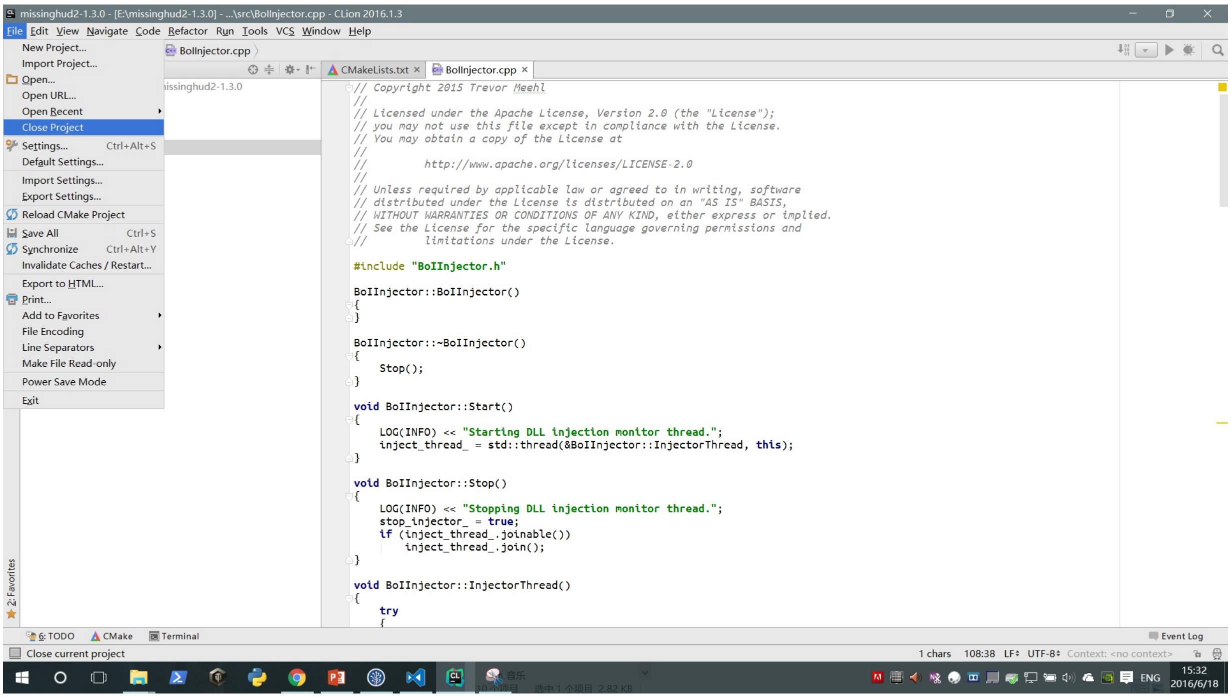
Before you start this guide, make sure you have

- Installed Clion correctly.
- Successfully get a student license. (Or you have the cracked version of the software)

Tool chain setup:

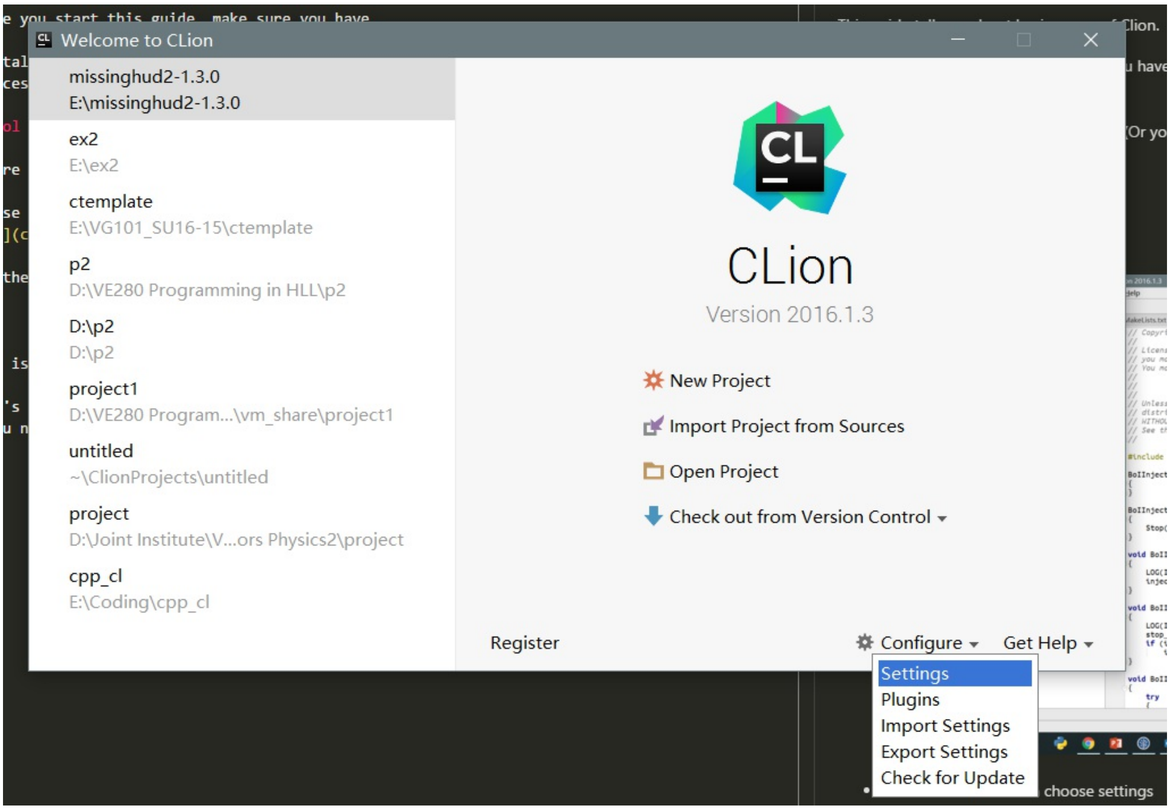
You are about to tell Clion where gcc is.

- Close your current project



```
// Copyright 2015 Trevor Meehl
//
// Licensed under the Apache License, Version 2.0 (the "License");
// you may not use this file except in compliance with the License.
// You may obtain a copy of the License at
//
//         http://www.apache.org/licenses/LICENSE-2.0
//
// Unless required by applicable law or agreed to in writing, software
// distributed under the License is distributed on an "AS IS" BASIS,
// WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
// See the License for the specific language governing permissions and
//         limitations under the License.

#include "BoIInjector.h"

BoIInjector::BoIInjector()
{
}

BoIInjector::~BoIInjector()
{
    Stop();
}

void BoIInjector::Start()
{
    LOG(INFO) << "Starting DLL injection monitor thread.";
    inject_thread_ = std::thread(&BoIInjector::InjectorThread, this);
}

void BoIInjector::Stop()
{
    LOG(INFO) << "Stopping DLL injection monitor thread.";
    stop_injector_ = true;
    if (inject_thread_.joinable())
        inject_thread_.join();
}

void BoIInjector::InjectorThread()
{
    try
    {
```

- In the welcome screen choose settings.



Welcome to CLion

missinghud2-1.3.0
E:\missinghud2-1.3.0

ex2
E:\ex2

ctemplate
E:\VG101_SU16-15\ctemplate

p2
D:\VE280 Programming in HLL\p2

D:\p2
D:\p2

project1
D:\VE280 Program...\vm_share\project1

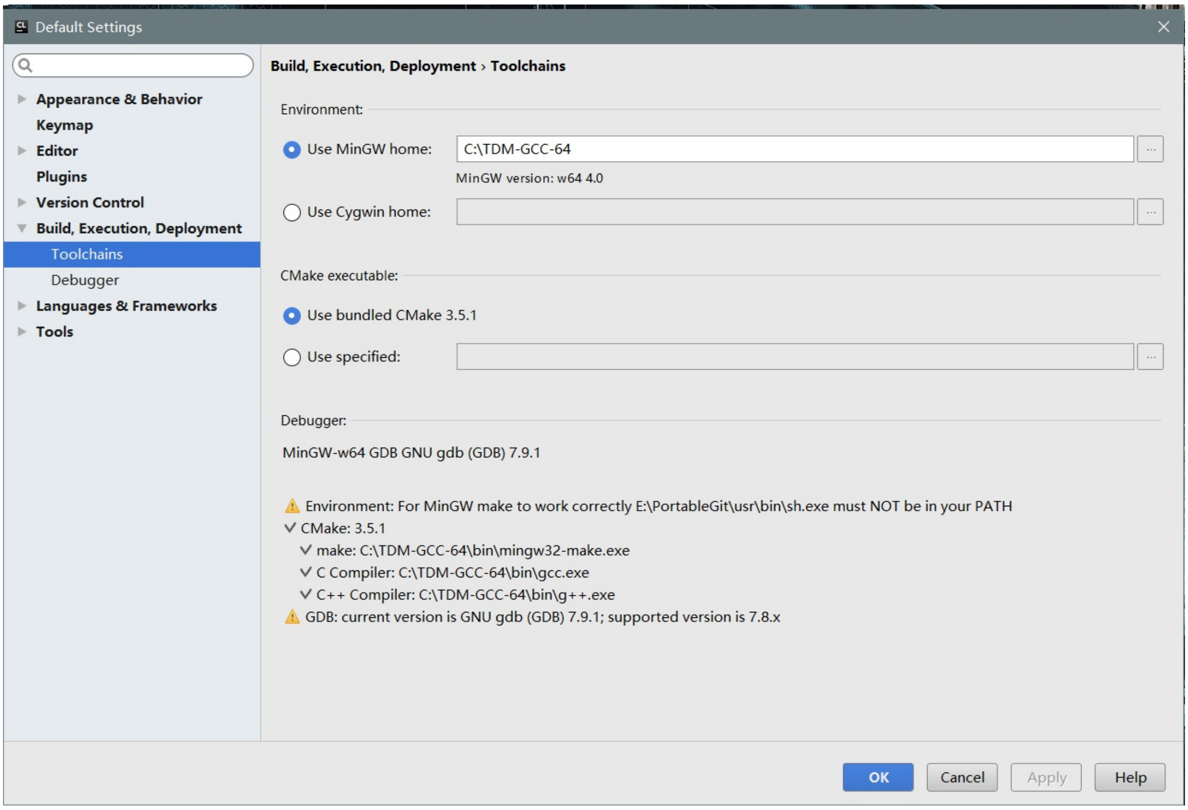
untitled
~\ClionProjects\untitled

project
D:\Joint Institute\V...ors Physics2\project

cpp_cl
E:\Coding\cpp_cl

CLion
Version 2016.1.3

- In the popup settings window, goto toolchain



Default Settings

Build, Execution, Deployment › Toolchains

Environment:

● Use MinGW home: C:\TDM-GCC-64

MinGW version: w64 4.0

○ Use Cygwin home:

CMake executable:

● Use bundled CMake 3.5.1

○ Use specified:

Debugger:

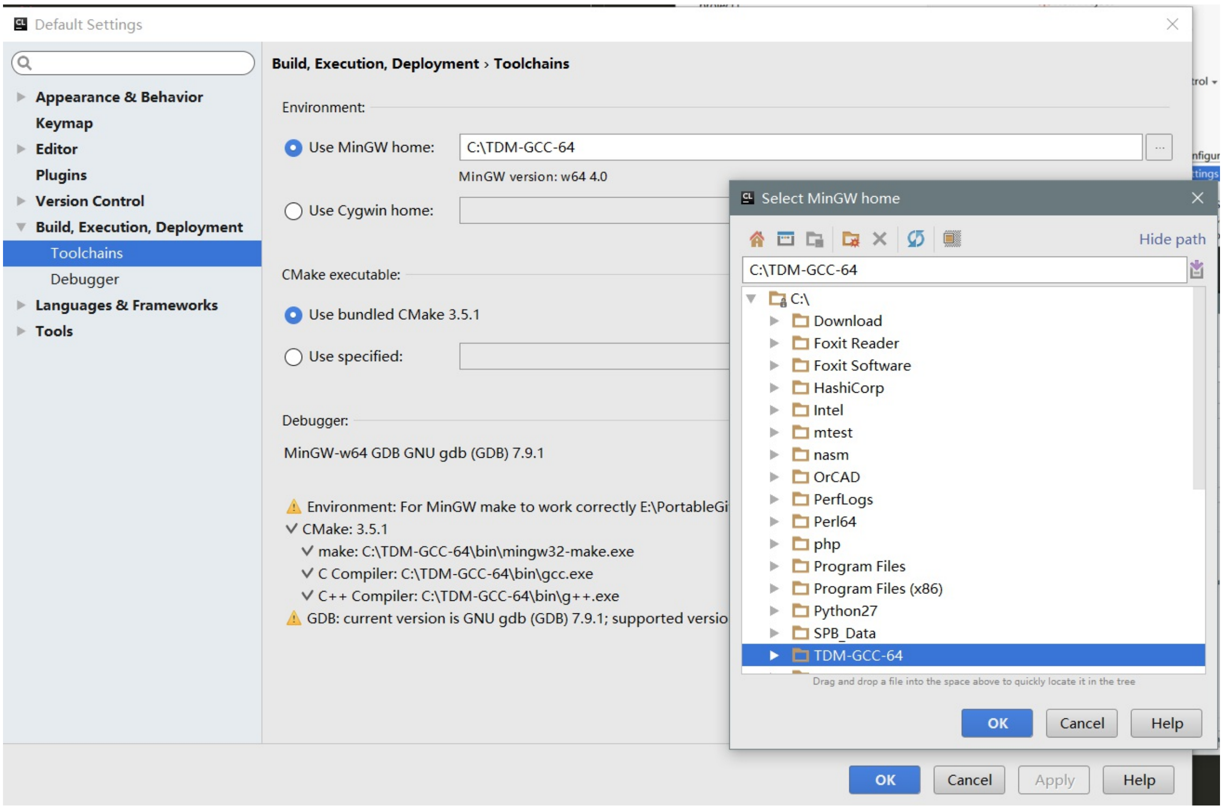
MinGW-w64 GDB GNU gdb (GDB) 7.9.1

⚠ Environment: For MinGW make to work correctly E:\PortableGit\usr\bin\sh.exe must NOT be in your PATH
✓ CMake: 3.5.1
  ✓ make: C:\TDM-GCC-64\bin\mingw32-make.exe
  ✓ C Compiler: C:\TDM-GCC-64\bin\gcc.exe
  ✓ C++ Compiler: C:\TDM-GCC-64\bin\g++.exe
⚠ GDB: current version is GNU gdb (GDB) 7.9.1; supported version is 7.8.x

- Keep others as they are, in Environment/Use MinGW home, chose your gcc installation dir

If you follow the instruction. The default dir should be (Mine is a little bit different):

```
C:\TDM-GCC-32
```



Build, Execution, Deployment › Toolchains

Environment:

● Use MinGW home: C:\TDM-GCC-64

MinGW version: w64 4.0

○ Use Cygwin home:

CMake executable:

● Use bundled CMake 3.5.1

○ Use specified:

Debugger:

MinGW-w64 GDB GNU gdb (GDB) 7.9.1

⚠ Environment: For MinGW make to work correctly E:\PortableGi

✓ CMake: 3.5.1

✓ make: C:\TDM-GCC-64\bin\mingw32-make.exe

✓ C Compiler: C:\TDM-GCC-64\bin\gcc.exe

✓ C++ Compiler: C:\TDM-GCC-64\bin\g++.exe

⚠ GDB: current version is GNU gdb (GDB) 7.9.1; supported versio

Select MinGW home: C:\TDM-GCC-64

- Wait some time for the software to verify your choice of location.

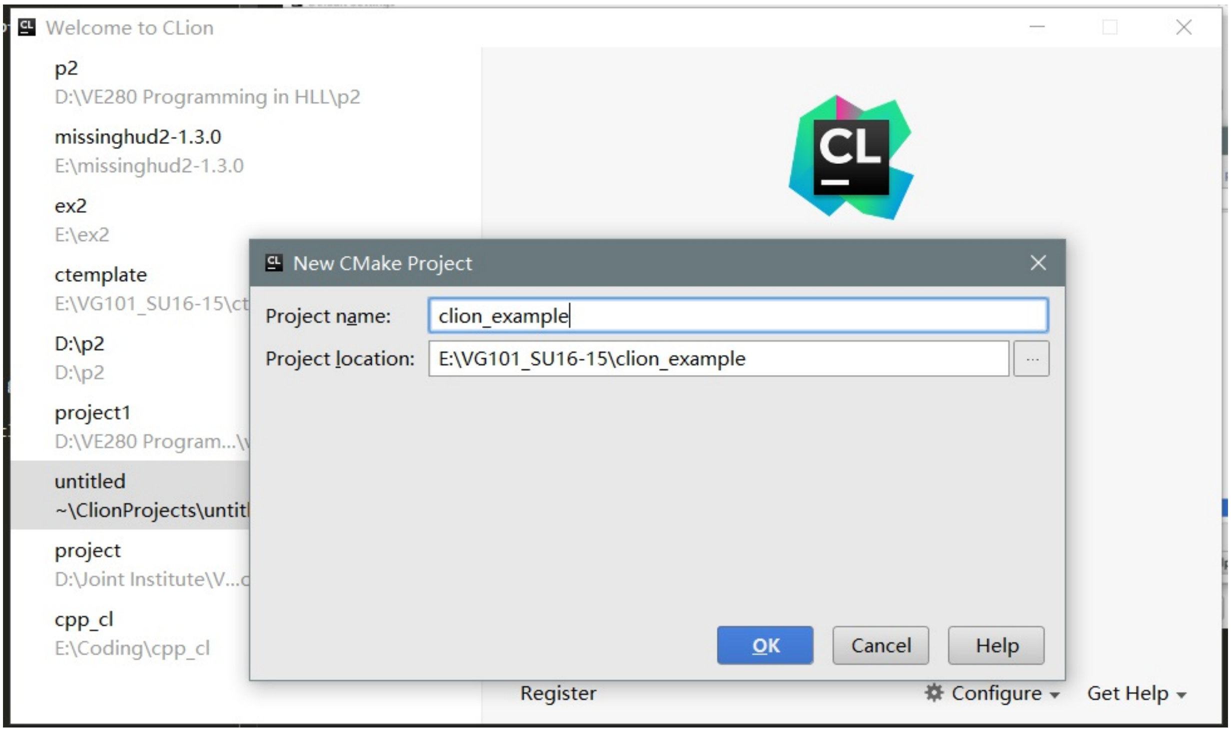
Normally you would get a warning, nerver mind. Just click OK

You have sucessfully set up gcc

The first project

- Click new project on the welcome screen. Enter your project name and place. Click OK.

  You must choose a place that does not contain non-english character in its path!



Welcome to CLion

p2
D:\VE280 Programming in HLL\p2

missinghud2-1.3.0
E:\missinghud2-1.3.0

ex2
E:\ex2

ctemplate
E:\VG101_SU16-15\ct

D:\p2
D:\p2

project1
D:\VE280 Program...\

untitled
~\ClionProjects\untit

project
D:\Joint Institute\V...

cpp_cl
E:\Coding\cpp_cl

New CMake Project

Project name: clion_example

Project location: E:\VG101_SU16-15\clion_example

- You will see a screen like this. Notice the words and progress bar in the red box.

  It indicates CLion is trying to process your code (And Environment). It takes time (depends on your CPU, and how large your program is). You can write code in this period. But you cannot compile anything yet.

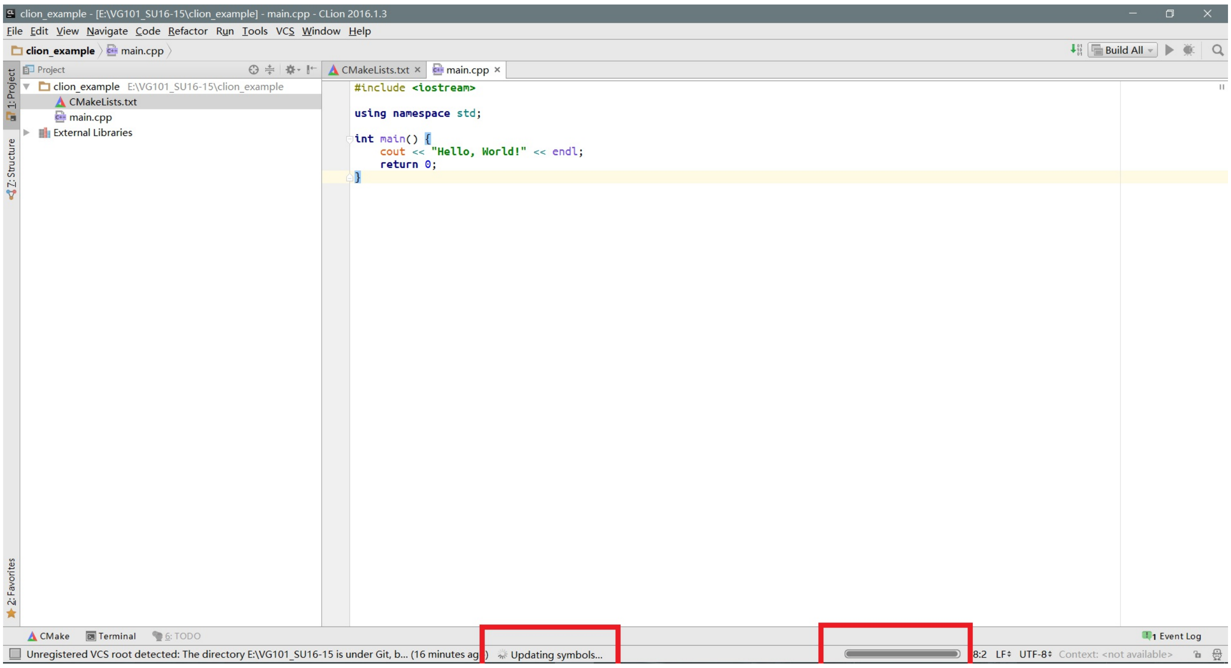
  This will happen only at:

  1. You just opened your project
  2. You added (deleted) files to compile in your project.

  Wait patiently. The time is worthy.

  By default, Clion considers itself a C++ IDE. Meaning it automatically generate a C++ sample file.

  Notice the file extention for this file is ".cpp".


clion_example - [E:\VG101_SU16-15\clion_example] - main.cpp - CLion 2016.1.3
File Edit View Navigate Code Refactor Run Tools VCS Window Help
clion_example
main.cpp
Build All
Project
clion_example E:\VG101_SU16-15\clion_example
CMakeLists.txt
main.cpp
External Libraries
CMakeLists.txt
main.cpp
1: Project
7: Structure
2: Favorites
CMake
Terminal
6: TODO
Event Log
8:2 LF UTF-8 Context: <not available>

```
#include <iostream>

using namespace std;

int main() {
    cout << "Hello, World!" << endl;
    return 0;
}
```

Unregistered VCS root detected: The directory E:\VG101_SU16-15 is under Git, b... (16 minutes ag

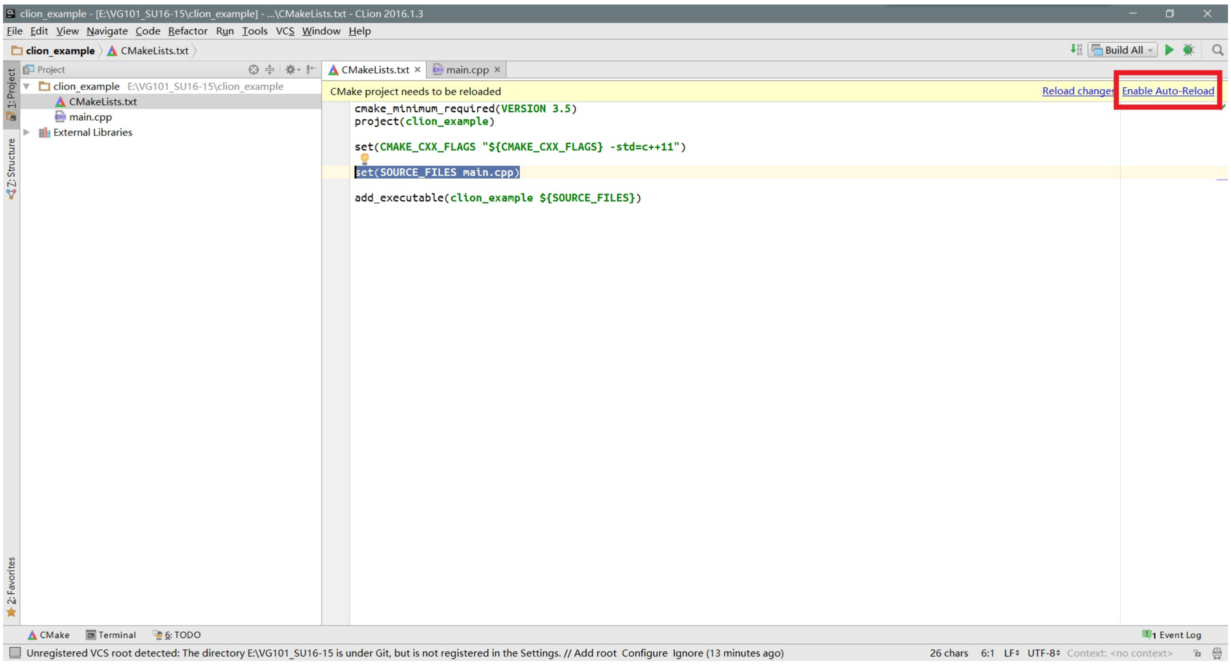
Updating symbols...

- Adding/Deleting file to compile

  We now need to delete the ".cpp" file. This is done by editing the "CMakeLists.txt".

  The "CMakeLists.txt" contains a list of files that will be compiled by the compiler.

  Double click "CMakeLists.txt" (This file has a triangle icon on the left) to open it.


clion_example - [E:\VG101_SU16-15\clion_example] - ...\CMakeLists.txt - CLion 2016.1.3
File Edit View Navigate Code Refactor Run Tools VCS Window Help
clion_example
CMakeLists.txt
Build All
Project
clion_example E:\VG101_SU16-15\clion_example
CMakeLists.txt
main.cpp
External Libraries
CMakeLists.txt
main.cpp
Reload changes
Enable Auto-Reload
1: Project
7: Structure
2: Favorites
CMake
Terminal
6: TODO
Event Log
26 chars 6:1 LF UTF-8 Context: <no context>

CMake project needs to be reloaded

```
cmake_minimum_required(VERSION 3.5)
project(clion_example)

set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -std=c++11")

set(SOURCE_FILES main.cpp)

add_executable(clion_example ${SOURCE_FILES})
```

Unregistered VCS root detected: The directory E:\VG101_SU16-15 is under Git, but is not registered in the Settings. // Add root Configure Ignore (13 minutes ago)

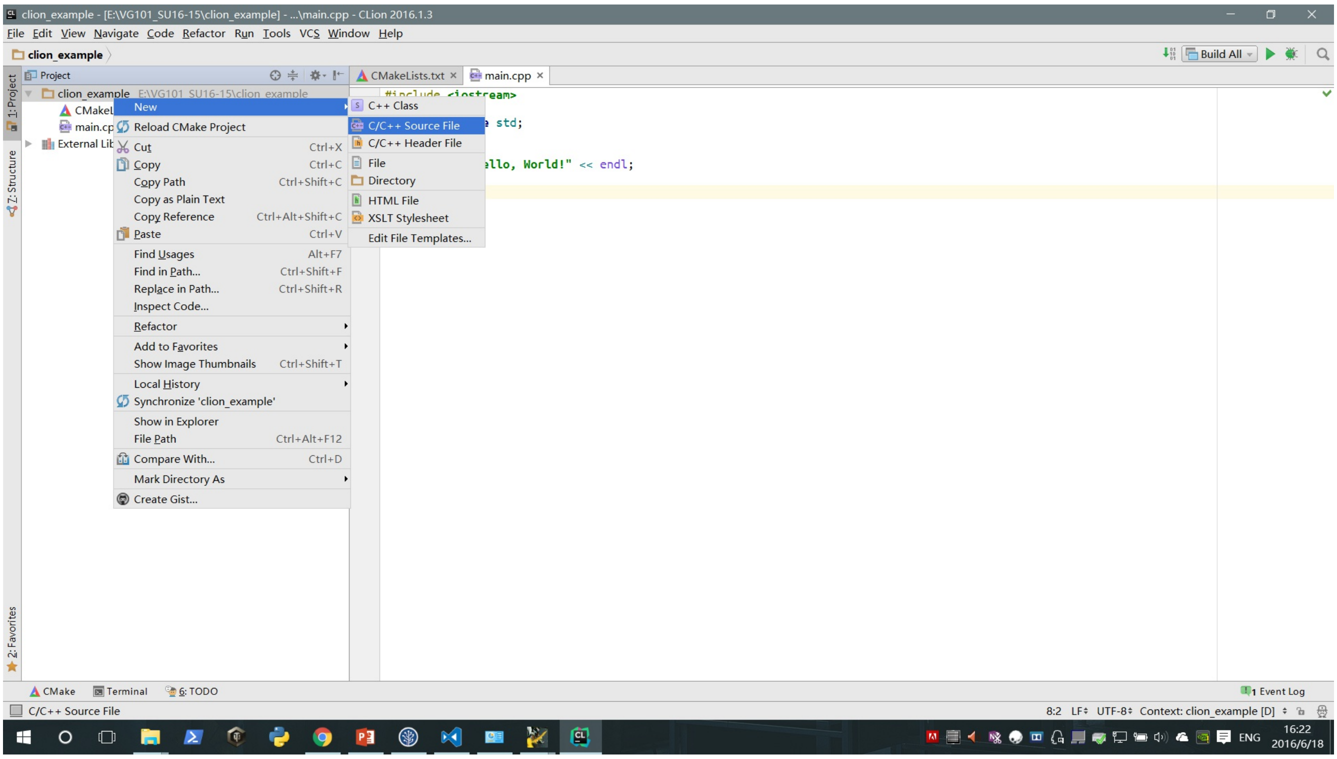
Change the line set(SOURCE_FILES main.cpp) into

```
set(SOURCE_FILS )
```

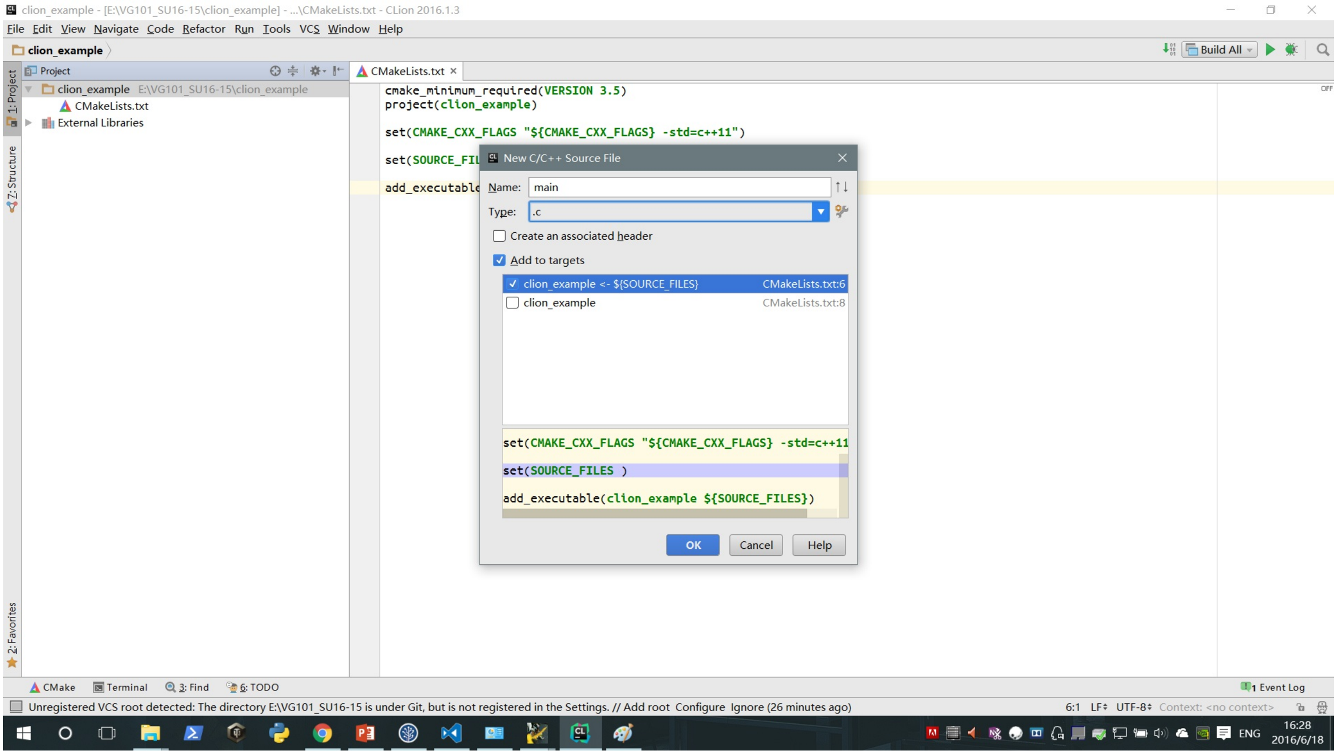
As the name suggests, this deletes main.cpp from the list of source files that will be compiled.

If the yello bar above appears, clik the red boxed one. Wait until Clion processed the change. Ignore any error. Now you should delete this file (physically, or in other words, delete in Windows).

Then we add our own C file. Right click on the project folder any click new/new C/C++ source file



In the pop up windows, in type select .C. This means it is a C Source file. Let's name it main. Click OK


clion_example - [E:\VG101_SU16-15\clion_example] - ...\CMakeLists.txt - CLion 2016.1.3
File Edit View Navigate Code Refactor Run Tools VCS Window Help
clion_example
Build All
Project
clion_example E:\VG101_SU16-15\clion_example
CMakeLists.txt
External Libraries
CMakeLists.txt
New C/C++ Source File
OK
Cancel
Help
CMake
Terminal
3: Find
6: TODO
Event Log
ENG
16:28
2016/6/18

```
cmake_minimum_required(VERSION 3.5)
project(clion_example)

set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -std=c++11")

set(SOURCE_FI

add_executable
```

Name: main

Type: .c

□ Create an associated header

☑ Add to targets

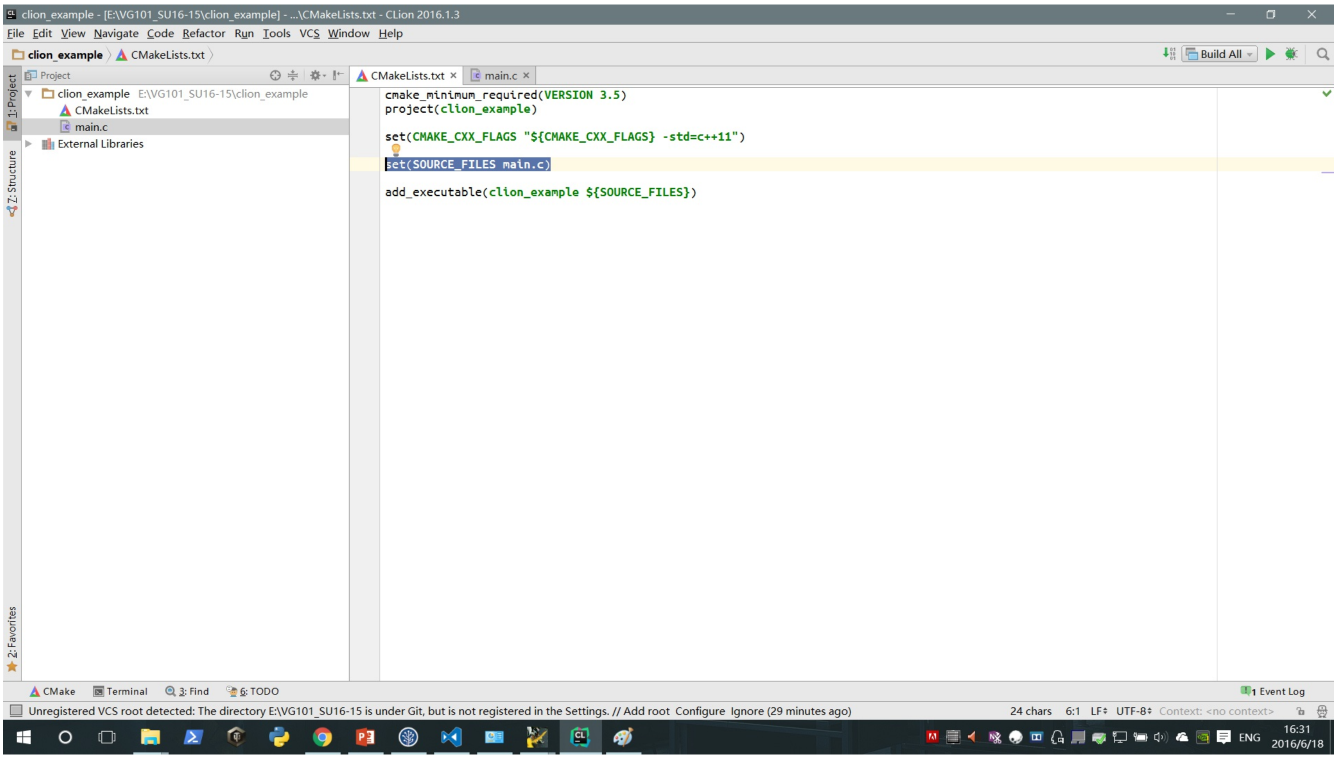
☑ clion_example <- ${SOURCE_FILES} CMakeLists.txt:6

□ clion_example CMakeLists.txt:8

```
set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -std=c++11

set(SOURCE_FILES )

add_executable(clion_example ${SOURCE_FILES})
```

Unregistered VCS root detected: The directory E:\VG101_SU16-15 is under Git, but is not registered in the Settings. // Add root Configure Ignore (26 minutes ago)

Check again the CMakeLists.txt, you see main.c in set(SOURCE_FILE main.c).


clion_example - [E:\VG101_SU16-15\clion_example] - ...\CMakeLists.txt - CLion 2016.1.3
File Edit View Navigate Code Refactor Run Tools VCS Window Help
clion_example
CMakeLists.txt
Build All
Project
clion_example E:\VG101_SU16-15\clion_example
CMakeLists.txt
main.c
External Libraries
CMakeLists.txt
main.c
CMake
Terminal
3: Find
6: TODO
Event Log
ENG
16:31
2016/6/18

```
cmake_minimum_required(VERSION 3.5)
project(clion_example)

set(CMAKE_CXX_FLAGS "${CMAKE_CXX_FLAGS} -std=c++11")

set(SOURCE_FILES main.c)

add_executable(clion_example ${SOURCE_FILES})
```

Unregistered VCS root detected: The directory E:\VG101_SU16-15 is under Git, but is not registered in the Settings. // Add root Configure Ignore (29 minutes ago)

24 chars 6:1 LF UTF-8 Context: <no context>

This means the sofware automatically added your new source file in to the list of file to compile

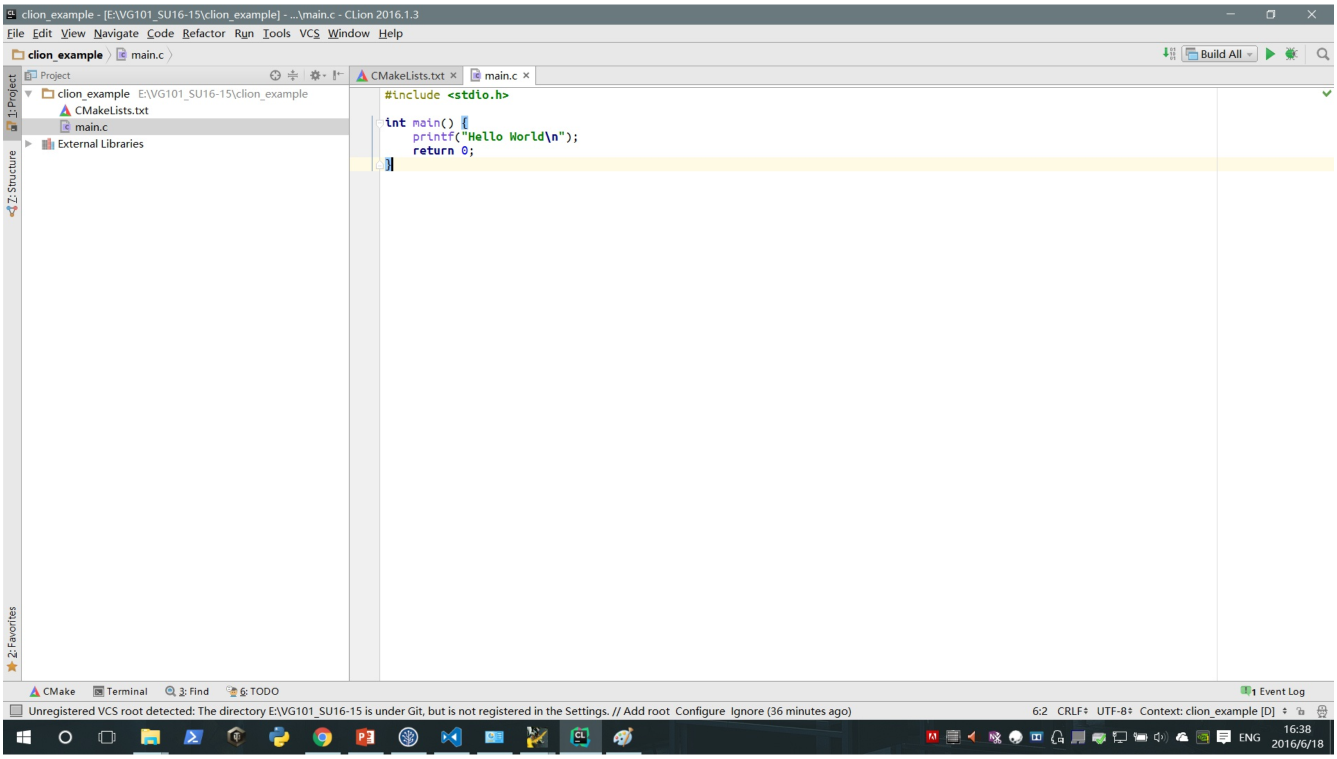
IMPORTANT

If you want to add existing file to your homework. Do this by

1. copy the file to your project directory
2. Add the file into set(SOURCE_FILE ...) in CMakeLists.txt

- Now you can open your main.c, write the hello world program.

```c
#include <stdio.h>

int main() {
    printf("Hello World\n");
    return 0;
}
```



clion_example - [E:\VG101_SU16-15\clion_example] - ...\main.c - CLion 2016.1.3

```c
#include <stdio.h>

int main() {
    printf("Hello World\n");
    return 0;
}
```

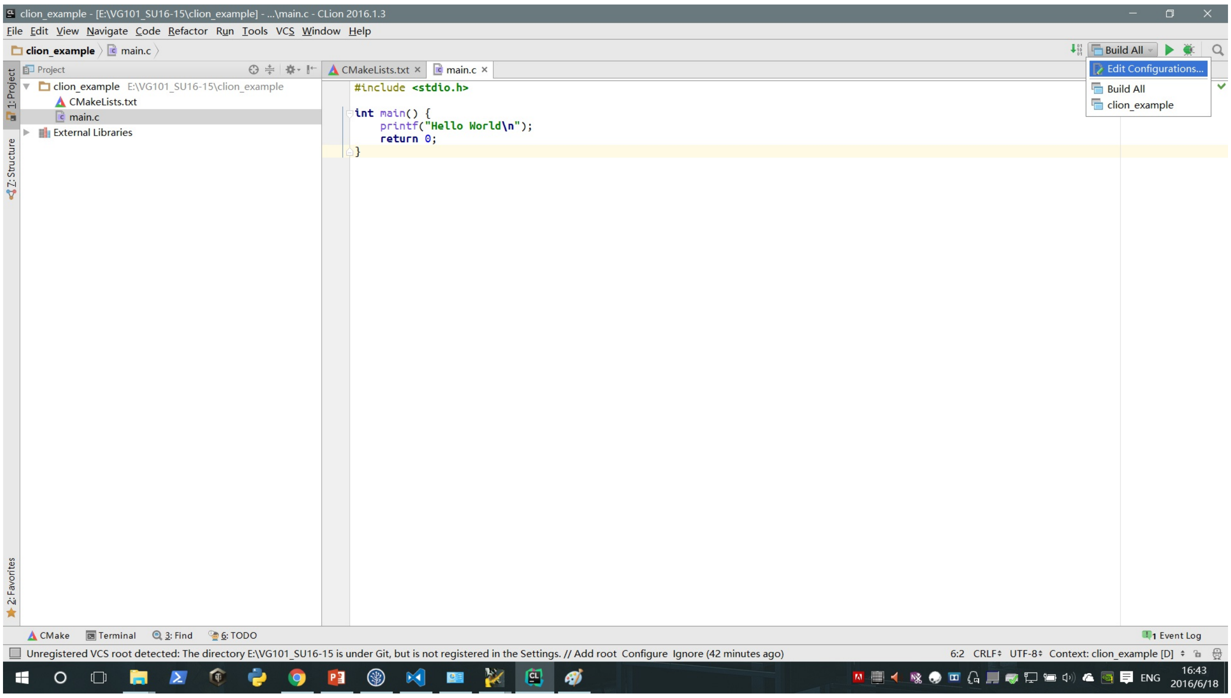
Unregistered VCS root detected: The directory E:\VG101_SU16-15 is under Git, but is not registered in the Settings. // Add root Configure Ignore (36 minutes ago)

Notice how CLion hightlights the "Hello world\n". It recogonized \n as a special character, thus notifying the programmer. These kind of design eventually cutdown mistakes.

- Tell software what to do after compiled your program

YOU ONLY NEED TO DO THIS ONCE FOR ONE HOMEWORK

Click Edit configuration:

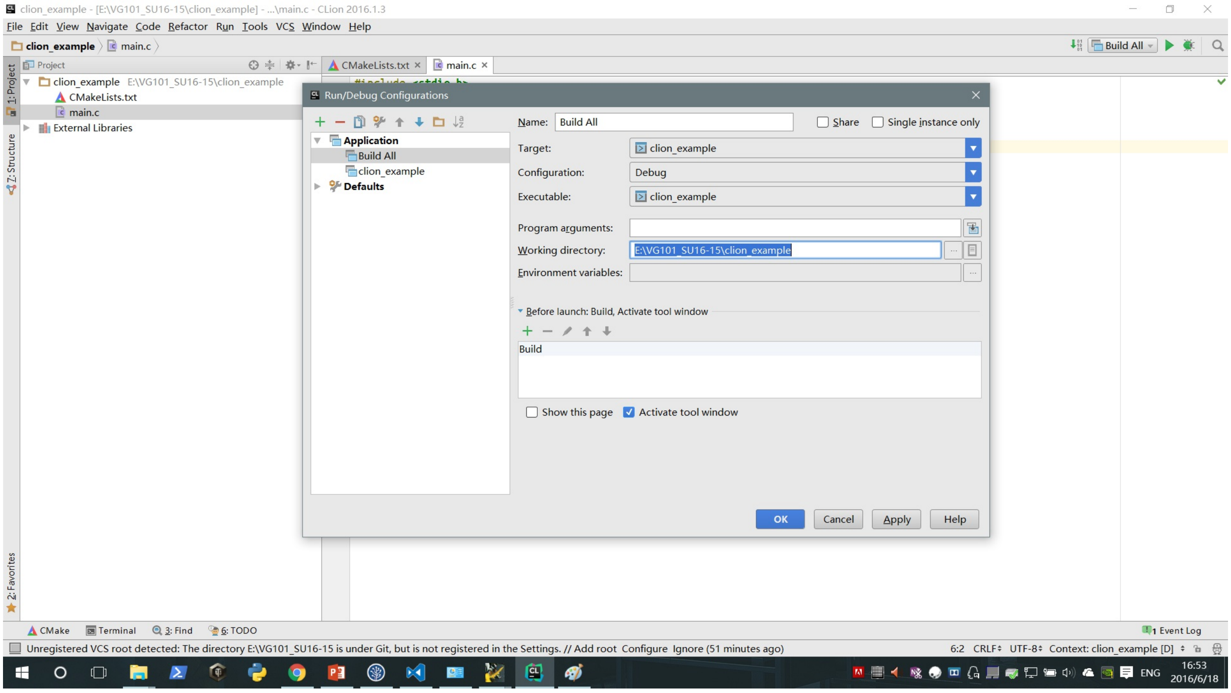

clion_example - [E:\VG101_SU16-15\clion_example] - ...\main.c - CLion 2016.1.3
File Edit View Navigate Code Refactor Run Tools VCS Window Help
Build All
Edit Configurations...
Build All
clion_example
Project
clion_example E:\VG101_SU16-15\clion_example
CMakeLists.txt
main.c
External Libraries
CMakeLists.txt
main.c
CMake
Terminal
3: Find
6: TODO
Event Log

```
#include <stdio.h>

int main() {
    printf("Hello World\n");
    return 0;
}
```

Unregistered VCS root detected: The directory E:\VG101_SU16-15 is under Git, but is not registered in the Settings. // Add root Configure Ignore (42 minutes ago)

Then input the following 2 things:

1. Set Executable to the only choice. This tells software what to run after compiling.
2. Set Working drectory to the directory of your project. This tells software where to run your code.


clion_example - [E:\VG101_SU16-15\clion_example] - ...\main.c - CLion 2016.1.3
File Edit View Navigate Code Refactor Run Tools VCS Window Help
Run/Debug Configurations
Application
Build All
clion_example
Defaults
OK
Cancel
Apply
Help

Name: Build All □ Share □ Single instance only

Target: clion_example

Configuration: Debug

Executable: clion_example

Program arguments:

Working directory: E:\VG101_SU16-15\clion_example

Environment variables:

Before launch: Build, Activate tool window

Build

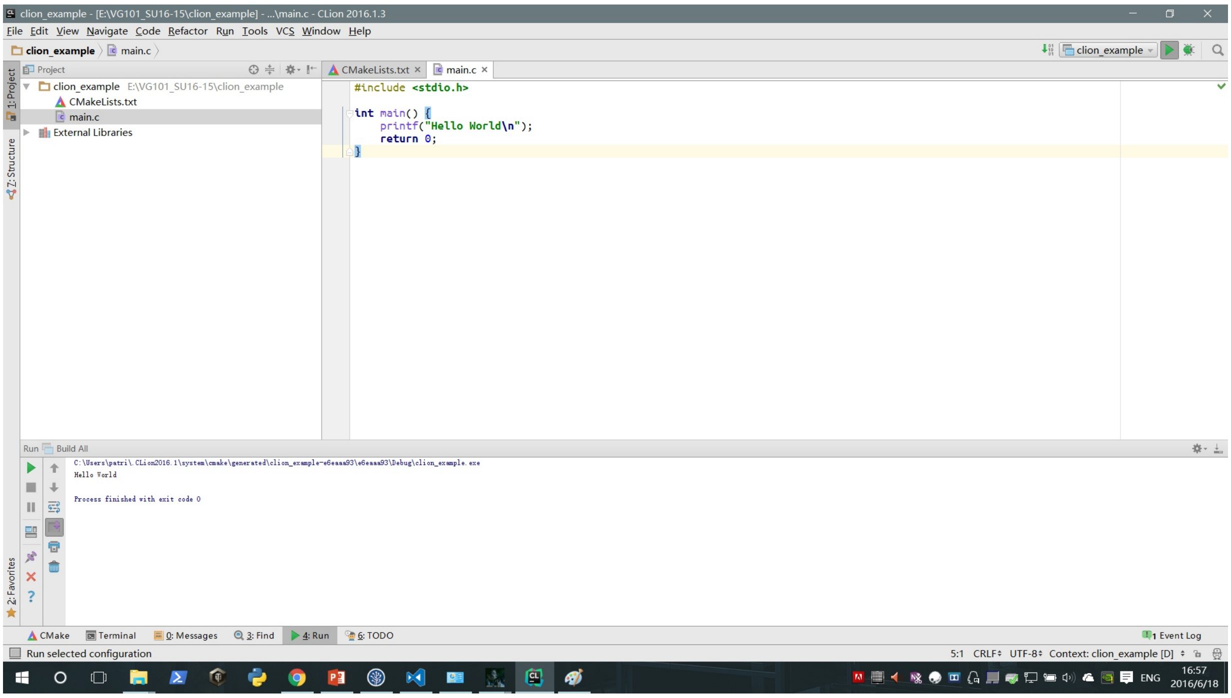
□ Show this page ☑ Activate tool window

Unregistered VCS root detected: The directory E:\VG101_SU16-15 is under Git, but is not registered in the Settings. // Add root Configure Ignore (51 minutes ago)

Click OK

- RUN, RUN, RUN!

Click the green arrow on the top right coner.



clion_example - [E:\VG101_SU16-15\clion_example] - ...\main.c - CLion 2016.1.3
File Edit View Navigate Code Refactor Run Tools VCS Window Help
clion_example
main.c
Project
clion_example E:\VG101_SU16-15\clion_example
CMakeLists.txt
main.c
External Libraries
CMakeLists.txt
main.c
Run
Build All
CMake
Terminal
0: Messages
3: Find
4: Run
6: TODO
1 Event Log
Run selected configuration
5:1 CRLF UTF-8 Context: clion_example [D]
ENG
16:57
2016/6/18

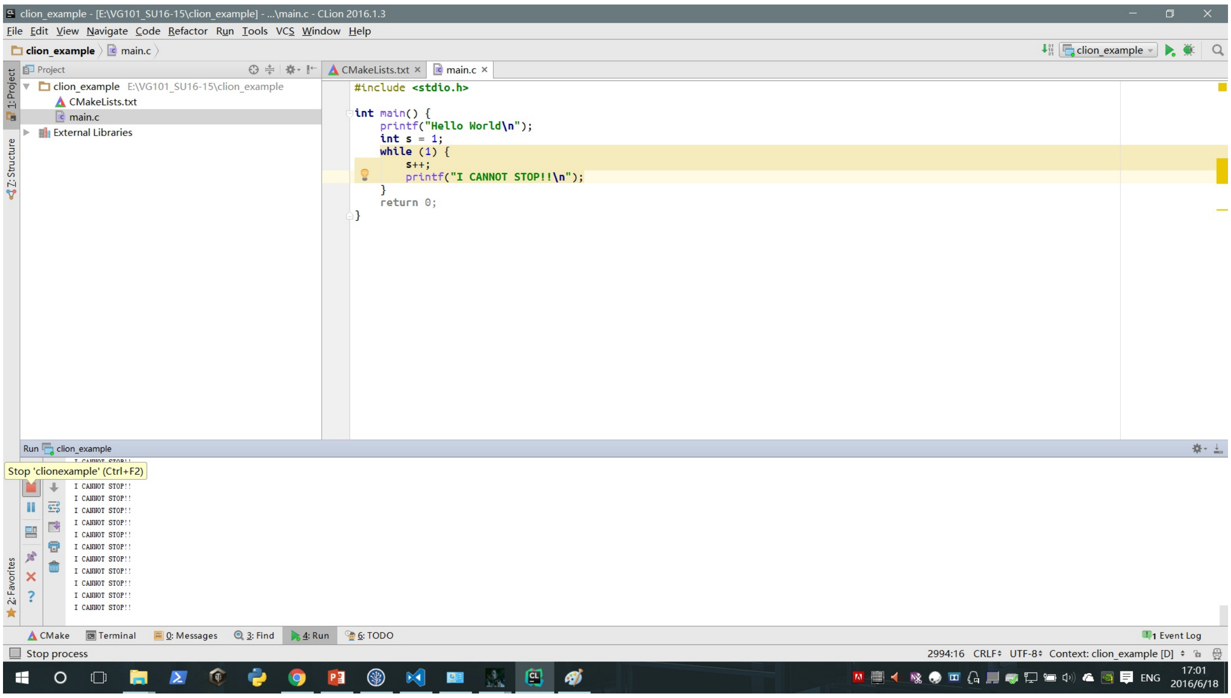
```c
#include <stdio.h>

int main() {
    printf("Hello World\n");
    return 0;
}
```

```
C:\Users\patri\.CLion2016.1\system\cmake\generated\clion_example-e6eaaa93\e6eaaa93\Debug\clion_example.exe
Hello World

Process finished with exit code 0
```

You will see your program's output in the pop up window on the bottom. If your program requires input from user, you can also type input into the box here. This allows you to write, compile and test your code without leaving the software, faster coding.

If you have written some wrong code, there will be compiler error on it.

Remember to end the running. Let's say you write an endless loop.



clion_example - [E:\VG101_SU16-15\clion_example] - ...\main.c - CLion 2016.1.3
Stop 'clionexample' (Ctrl+F2)
Stop process

```c
#include <stdio.h>

int main() {
    printf("Hello World\n");
    int s = 1;
    while (1) {
        s++;
        printf("I CANNOT STOP!!\n");
    }
    return 0;
}
```

```
I CANNOT STOP!!
I CANNOT STOP!!
I CANNOT STOP!!
I CANNOT STOP!!
I CANNOT STOP!!
I CANNOT STOP!!
I CANNOT STOP!!
I CANNOT STOP!!
I CANNOT STOP!!
I CANNOT STOP!!
I CANNOT STOP!!
I CANNOT STOP!!
```

You can click the red button to stop the execution. If you forgot to do that. Since your previous executable is still running, you will not be able to generate a new new executable to overwrite the previous one. Then you will get a compiler error.

Also notice CLion colors the content in while as yellow and the return 0; as gray. Yellow means warning. If you put your mouse onto the the yellow block on the right edge of the screen. It will say "endless loop". And the gray code are the code that will never get a chance to execute.

New programmers do accidentally write all sorts of endless loop, or accidentally wrong condition statements that some of the code will never get to execute. The software checks it, tries to warn you when you are coding this area, when you still remember what you have coded.

Another common mistake is missing semicolumns ";". Intentionally write a line without semicolumn, see how the software tell you.

Just another reason to use CLion.

Other questions:

- How do I add a header file?

  You do exactly as adding a ".c" file, expect you select "New\C/C++ Header"