D-SOFT IoTLab C++ Training Material

Written by: Phuoc NGUYEN March 22nd 2022 Revision 1.0

Prerequisite	2
CMake - C++ Class - Unit Test with Google	2
Task 1: getting familiar with CMake	2
Task 2: write two types of Png classes in C++	3
Task 3: write Unit Test	5
References	6

Prerequisite

All exercises/tasks need to conform to a coding style.

In our case, we will follow Google Coding Style for C++.

In the case of using any external libraries, you need to install it on your own.

However, you will be asked about the installation procedure, so take note of those steps.

CMake - C++ Class - Unit Test with Google

Deadline: you are required to submit your solution before 12PM on Thursday 24th 2022.

- Your code needs to be compiled successfully when review
- You have to answer every aspect in your code
- You can discuss the approach with your fellows, but each needs to write and understand the code clearly.
 - You don't have any grades here, you are doing this for your own benefit.
 - In case of getting stuck (even after consulting every material), you can come and ask me in person

Task 1: getting familiar with CMake

You are given a sample program with one CMake file. Run it and understand its basic content. You are required to write a program which:

- utilizes multiple source files (as a library) and
- uses CMake to compile it

Task 2: write two types of Png classes in C++

In this task, you are required to write to C++ classes with a member function that opens a PNG image file and prints the image info (such as height and width in pixels).

One class is named `Pngpplmage` and the other is `Lodepnglmage`.

- Each class must have the following function:
 - bool open_image(std::string image_path);
 - Return true if the image can be opened successfully, using its corresponding library
 - Return false otherwise
- 'Pngpplmage' must use libpng++ library
- `Lodepnglmage` must use LodePNG library
- Hint: to check for file existence, you can use **filesystem** library
- Feel free to add any other functions if needed

Write a sample program to invoke those two classes. Sample run:

\$./app_images

Sample Test Program for Pngpplmage and Lodepnglmage.

Use class 'Pngpplmage' to read an image file.

W: 160 H: 160

Test of PngppImage is successful

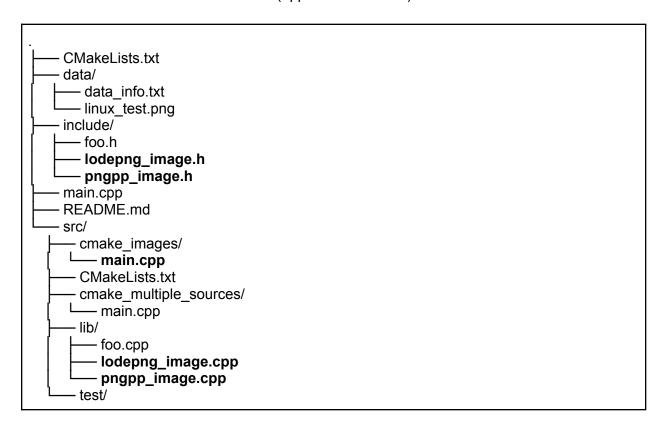
Use class 'LodepngImage' to read an image file.

W: 160 H: 160

Test of LodepngImage is successful

Directory structure:

- NOTE: you need to adapt in case of adding a new library, but basically, you need to conform to the following convention:
 - Header of library must be in **include**/
 - Source file of library must be in lib/
 - Test file must be in **test**/ (applicable to task 3)



Task 3: write Unit Test

Write a unit test to test for member function implementation of the two classes in task 2. You will use **Google Test** to write the unit tests.

Sample run:

References

- Cmake
 - o https://cmake.org/cmake/help/latest/
- Google Test
 - https://github.com/google/googletest
- Filesystem
 - o https://en.cppreference.com/w/cpp/filesystem
- Pngpp
 - o https://www.nongnu.org/pngpp/doc/0.2.9/
- Lodepng
 - o https://github.com/lvandeve/lodepng
- Google Style for C++
 - https://google.github.io/styleguide/cppguide.html#Naming
- TBA