WrpBase

IOTPRJ – 2020/03/21

History

Revision	Author	Date	Comments
1.0	nguyenhtm	2019/10/06	Initial version
1.1	nguyenhtm	2020/03/21	Update setup guidelines

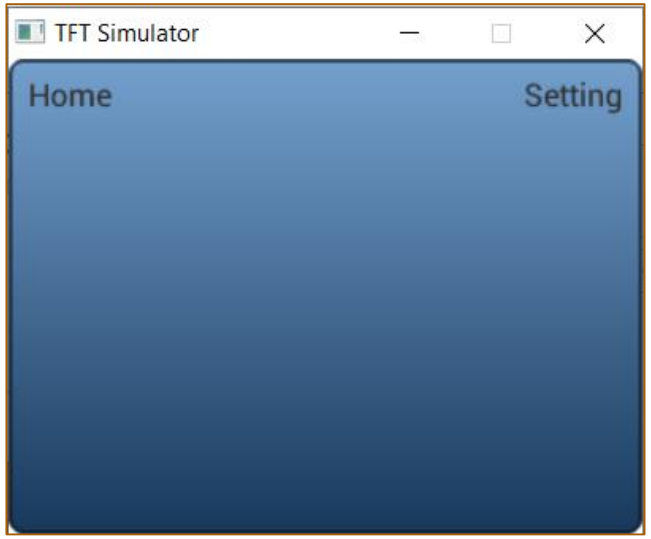
Overview

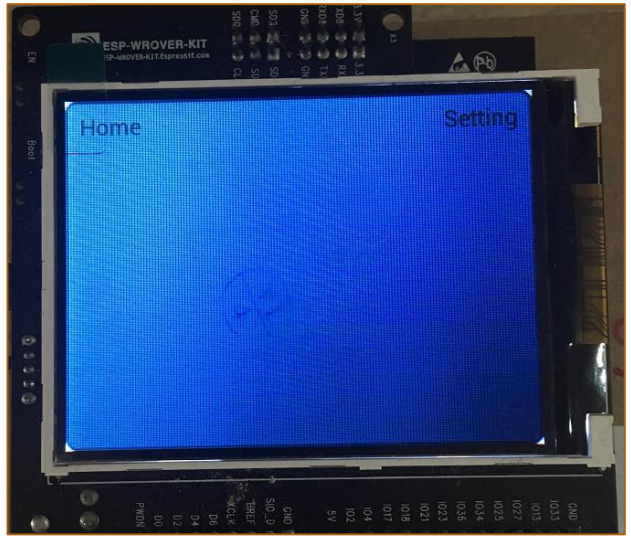
Provide a framework for developers to easily create GUI applications in IOT area with ESP32 chip

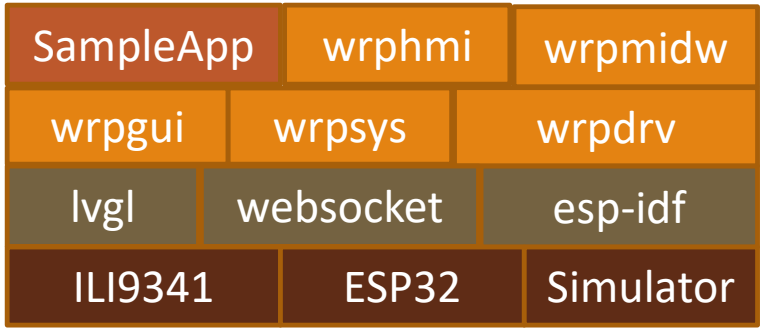
Sample App

The Sample App can run with ESP-WROVER-KIT V3 and on Windows platform using SDL Simulator

Video: https://www.youtube.com/watch?v=e4u56qCMuuE







Setup

D:\iotprj\wrpbase

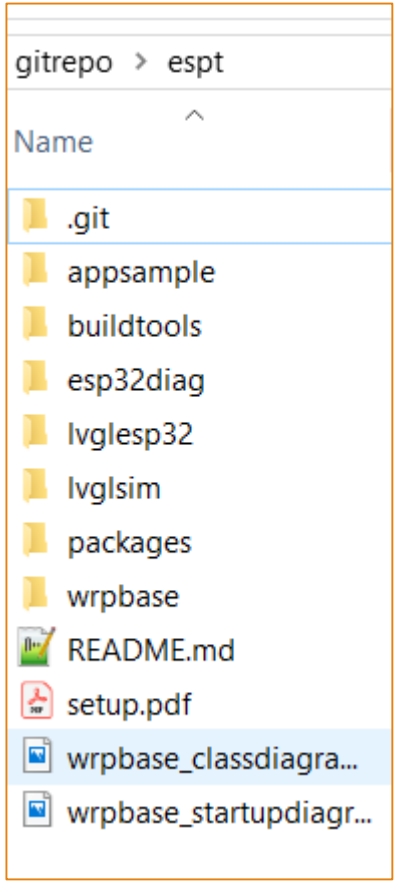
- D:\iotprj: git clone --recursive https://github.com/nguyenhtm/espt
- Folder Structures: D:\iotprj**appsample**, D:\iotprj**lvglesp32**, D:\iotprj**lvglsim**, D:\iotprj**wrpbase**

D:\iotprj\buildtools

- D:\iotprj\buildtools\mingw-w64: download mingw-w64-install.exe at http://mingw-w64.org/doku.php/download
- D:\iotprj\buildtools\msys32: download esp32_win32_msys2_environment_and_toolchain-20181001.zip at https://docs.espressif.com/projects/esp-idf/en/v3.3/get-started/windows-setup.html
- D:\iotprj\buildtools\eclipse: download eclipse-cpp-2019-03-R-win32-x86_64.zip at https://www.eclipse.org/downloads/packages/file/55067

D:\iotprj\packages

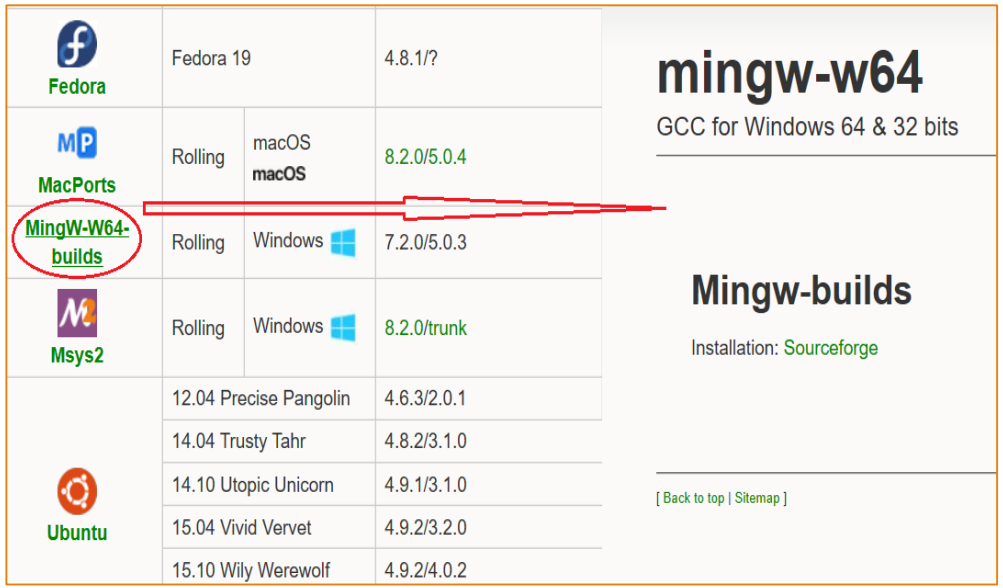
- D:\iotprj\packages\esp-idf: git clone --recursive https://github.com/espressif/esp-idf.git
- D:\iotprj\packages\lvgl: git clone --recursive https://github.com/littlevgl/lvgl.git
- D:\iotprj\packages\pc_simulator_sdl_eclipse: git clone --recursive https://github.com/littlevgl/pc_simulator_sdl_eclipse.git
- D:\iotprj\packages\esp32_ili9341: git clone --recursive https://github.com/littlevgl/esp32_ili9341.git
- D:\iotprj\packages\SDL2-2.0.5: download the SDL2-devel-2.0.5-mingw.tar.gz file at https://www.libsdl.org/release
- D:\iotprj\packages\mongoose: git clone https://github.com/cesanta/mongoose
- Create D:\iotprj\packages\mongoose_lib and copy 2 D:\iotprj\packages\mongoose\mongoose.c and mongoose.h files to it

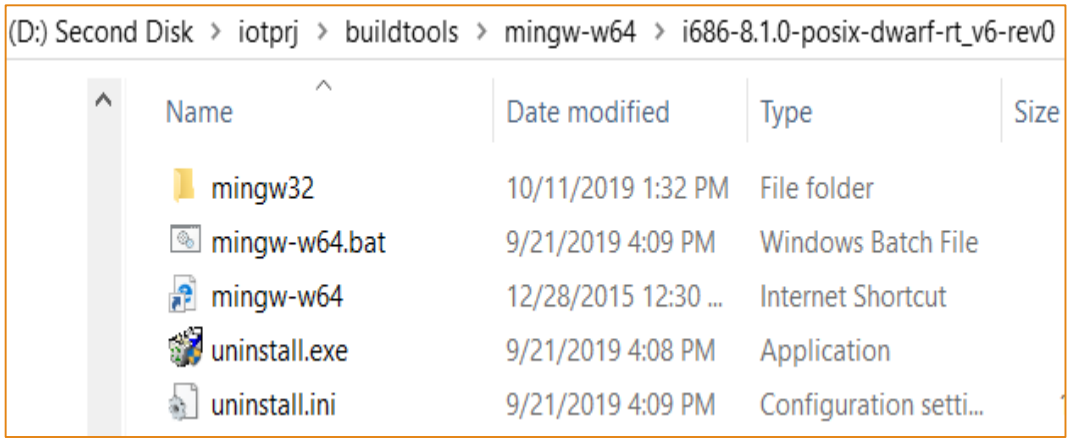


BuildTools

Mingw64

- GCC for Windows 32/64 bit which is used to compile source code for simulation
- Download mingw-w64-install.exe at http://mingw-w64.org/doku.php/download and install in buildtools folder



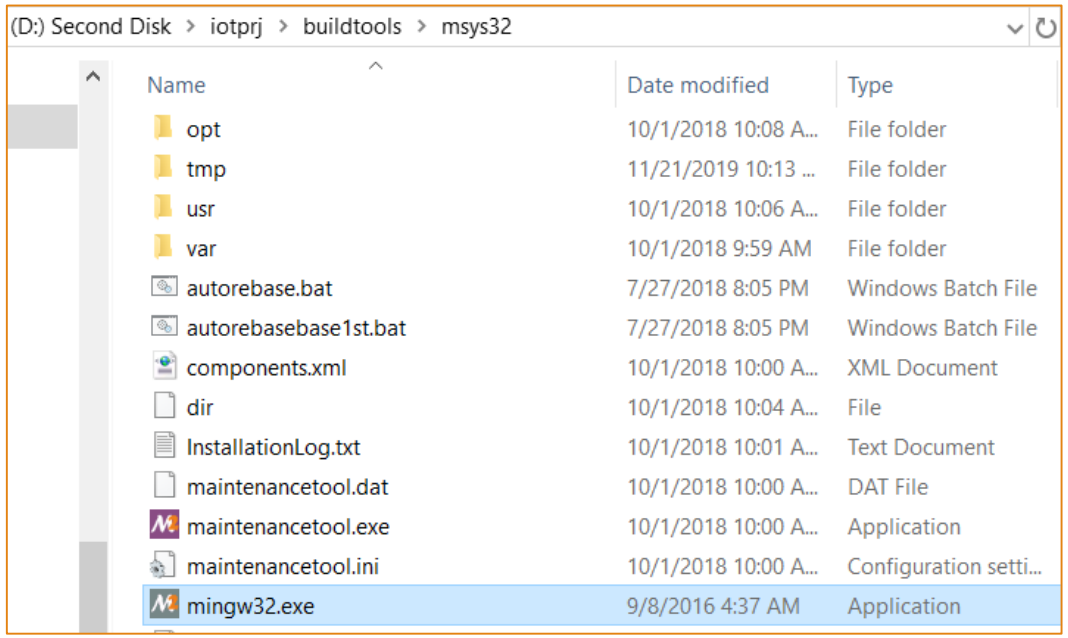


GCC install path: D:\iotprj\buildtools\mingw-w64\i686-8.1.0-posix-dwarf-rt_v6-rev0\mingw32\bin

BuildTools 2

Msys32 with ESP32

- GCC for ESP32 which is used to compile source code for PICO target
- Download esp32_win32_msys2_environment_and_toolchain-20181001.zip at https://docs.espressif.com/projects/esp-idf/en/v3.3/get-started/windows-setup.html and extract at buildtools folder

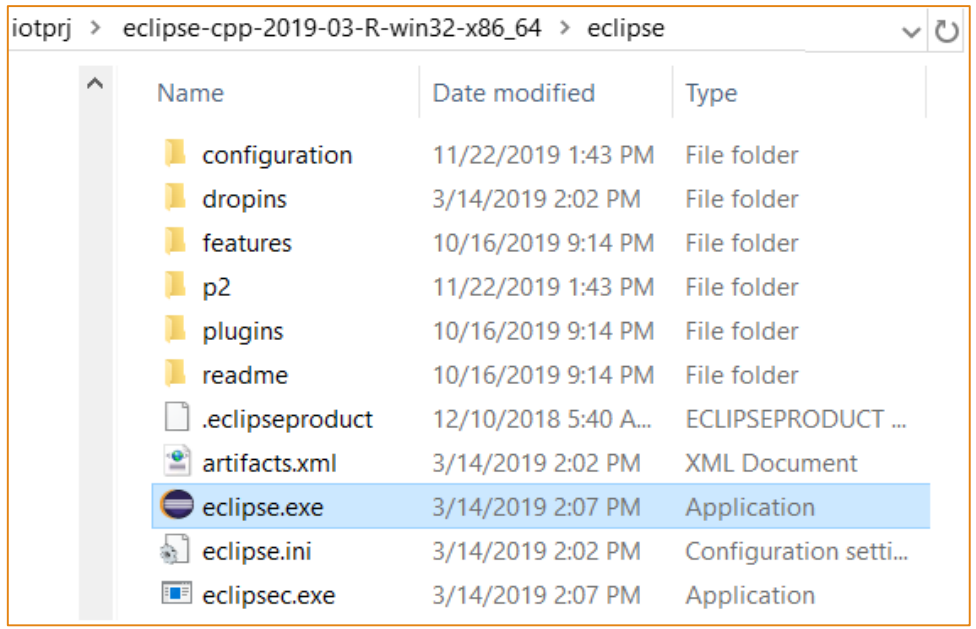


GCC install path:
D:\iotprj\buildtools\msys32\opt\xtensa-esp32-elf\bin

BuildTools 3

Eclipse

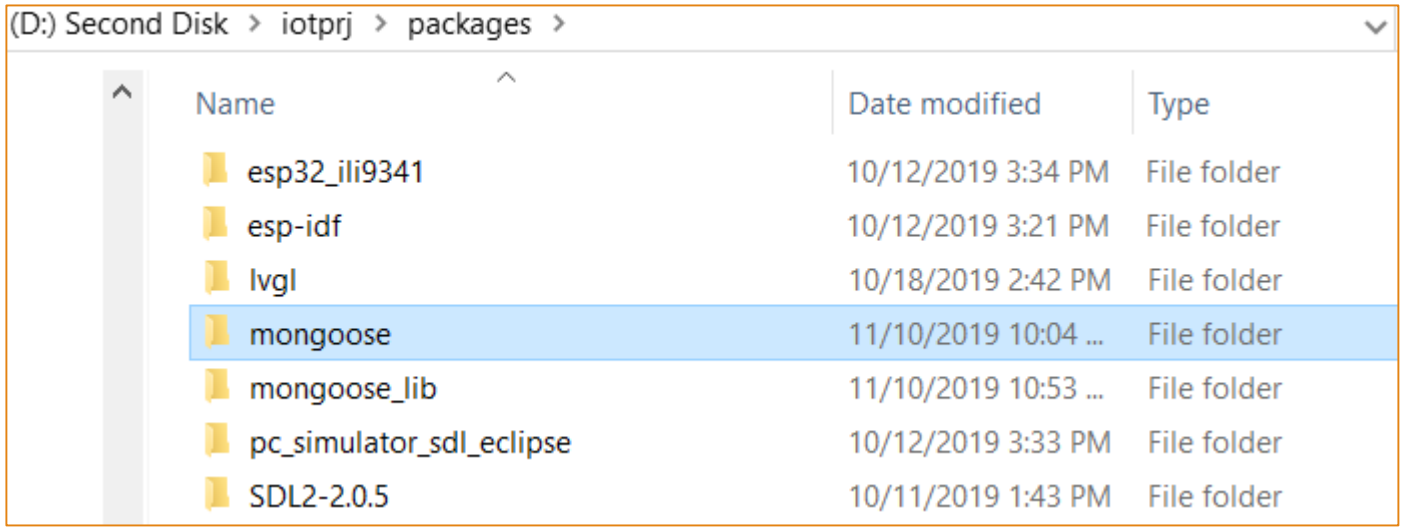
- Editor can be editor and configured to use GCC to build the sample app on PICO target and Windows
- Download eclipse-cpp-2019-03-R-win32-x86_64.zip at https://www.eclipse.org/downloads/packages/file/55067 and extract at buildtools folder



Eclipse install path:
D:\iotprj\buildtools\eclipse-cpp-2019-03-R-win32-x86_64\eclipse

Packages

- The folder used to store open sources like lvgl, mongoose or esp-idf framework. Refer to Setup slide to download
- In case of mongoose, after downloading, create **mongoose_lib** folder and copy **mongoose.c** and **mongoose.h** files to it

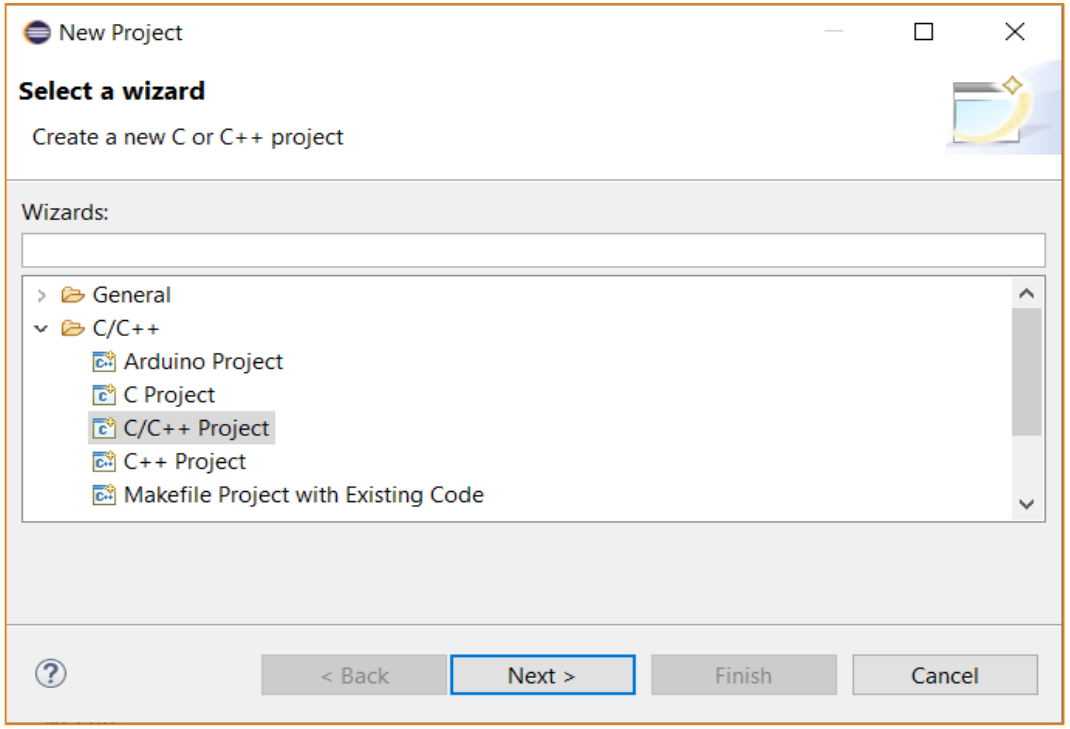


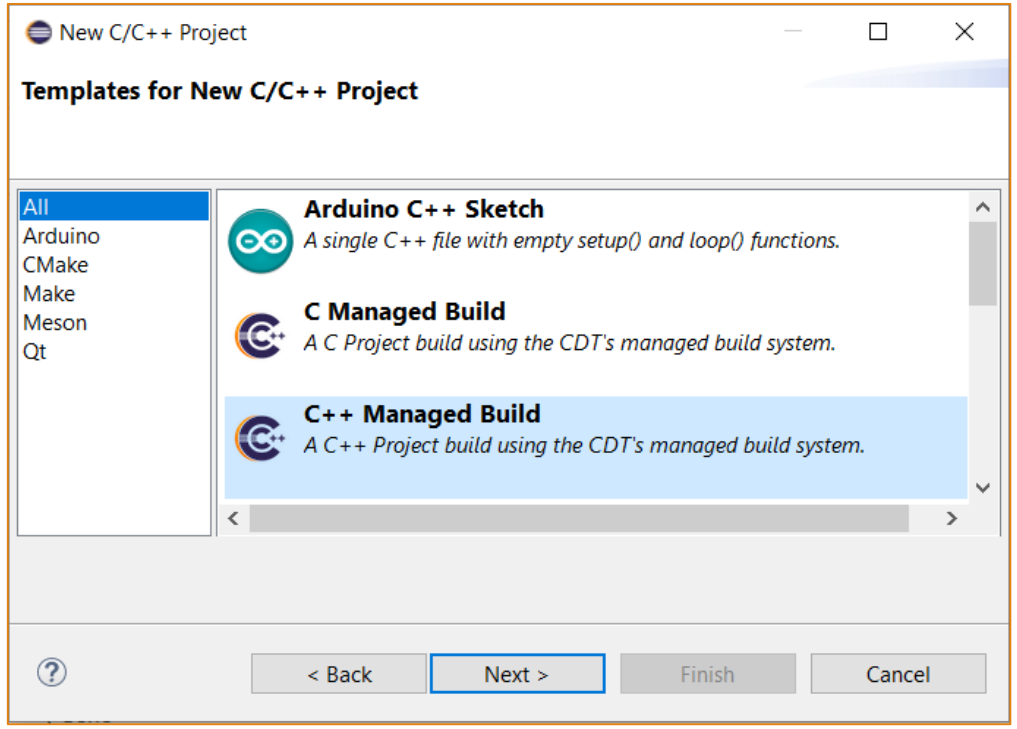
Notes

- Set IDF_PATH: echo 'export IDF_PATH="D:/iotprj/packages/esp-idf"' >> $HOME/.bash_profile
  - When open MSYS32 execute file, it will use this path as default.
- Generate partition table:
  - Partition table should not same name as project name otherwise same .bin is generated
  - Generate: python $IDF_PATH/components/partition_table/gen_esp32part.py projectname_partition_table.bin
  - Flash: python $IDF_PATH/tools/idf.py partition_table-flash => enter to lvglesp32\build\partition_table
- Execute the command to setup required software:
  - D:/iotprj/buildtools/msys32/mingw32/bin/python.exe -m pip install --user -r D:/iotprj/packages/esp-idf/requirements.txt

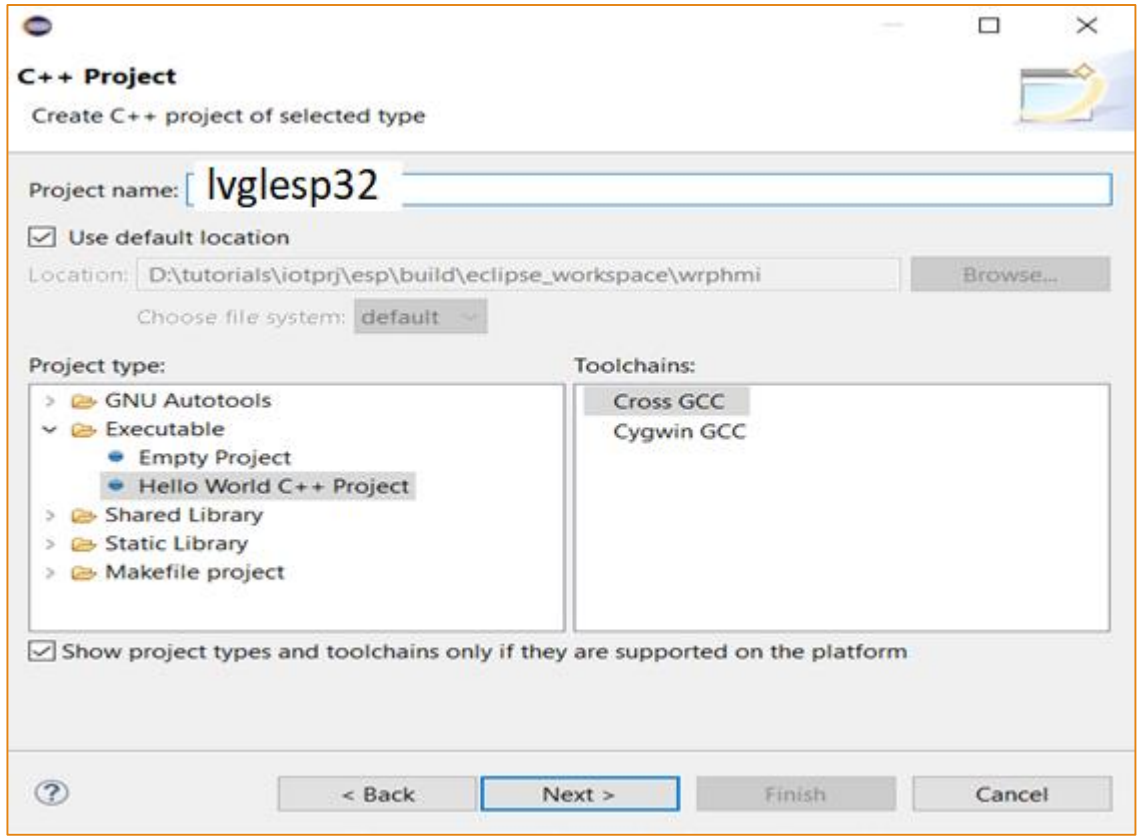
Eclipse C++ Project 1

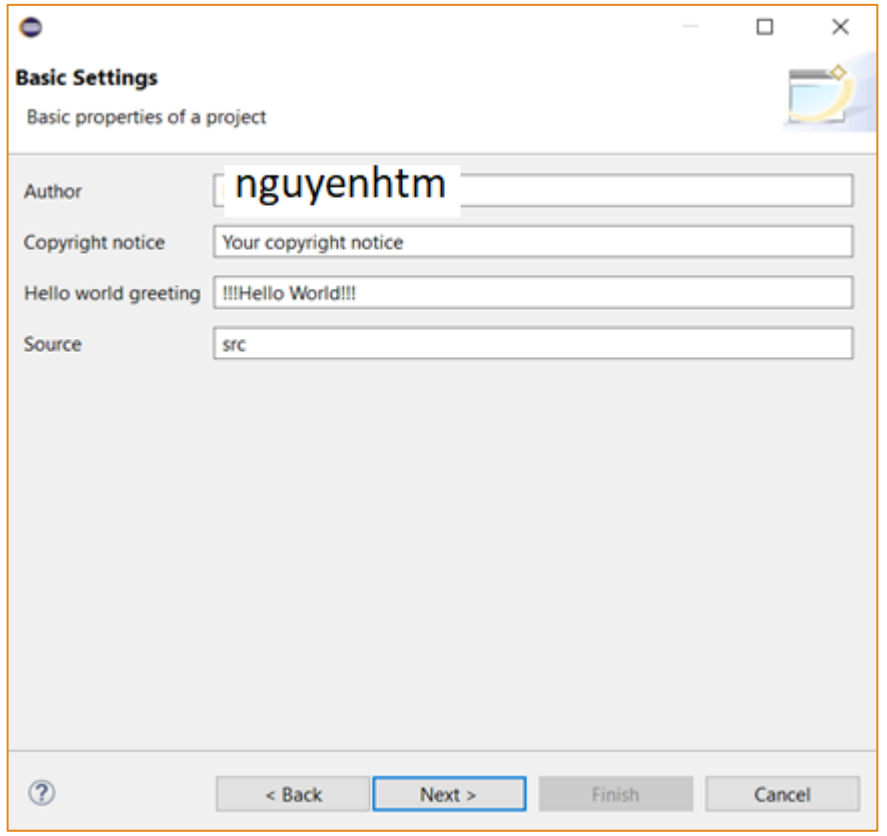
- Create 2 Eclispse C++ projects: one for simulation and one for ESP32 with LCD
- Both demo how to setup working environment and how to use open sources like lvgl, mongoose,…



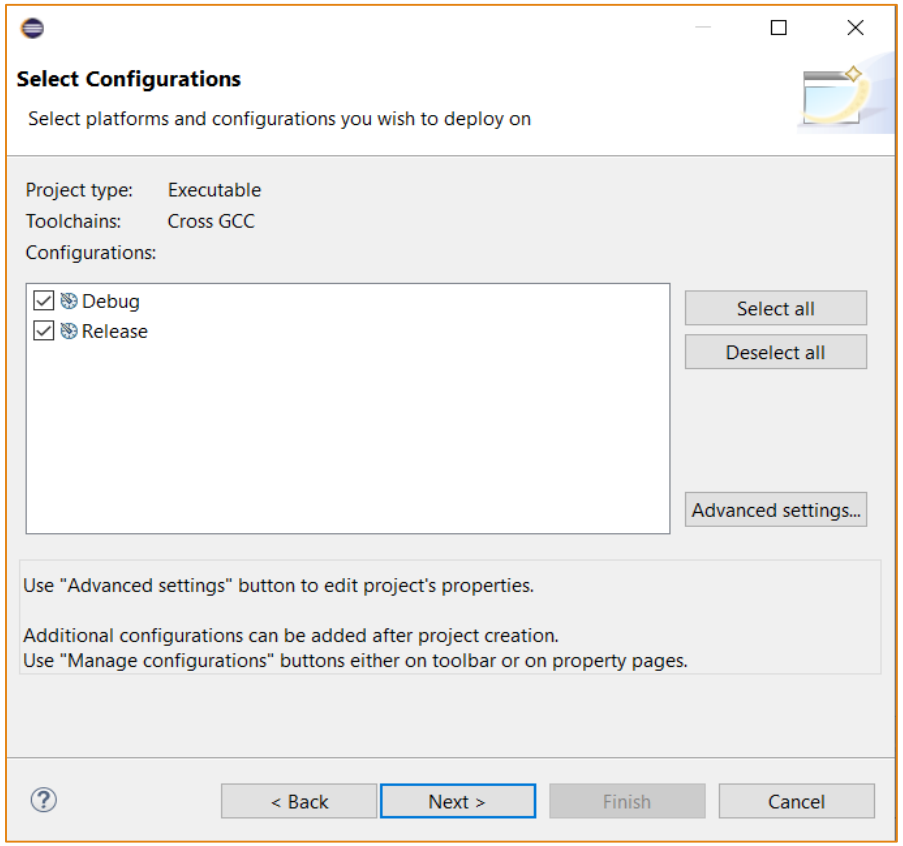


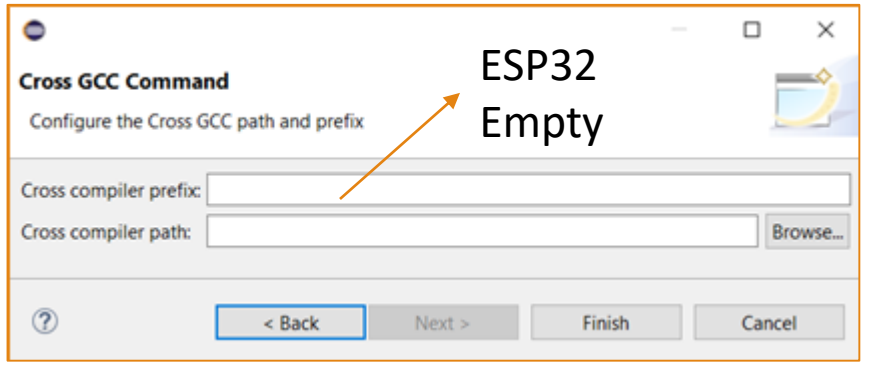
Eclipse C++ Project 2



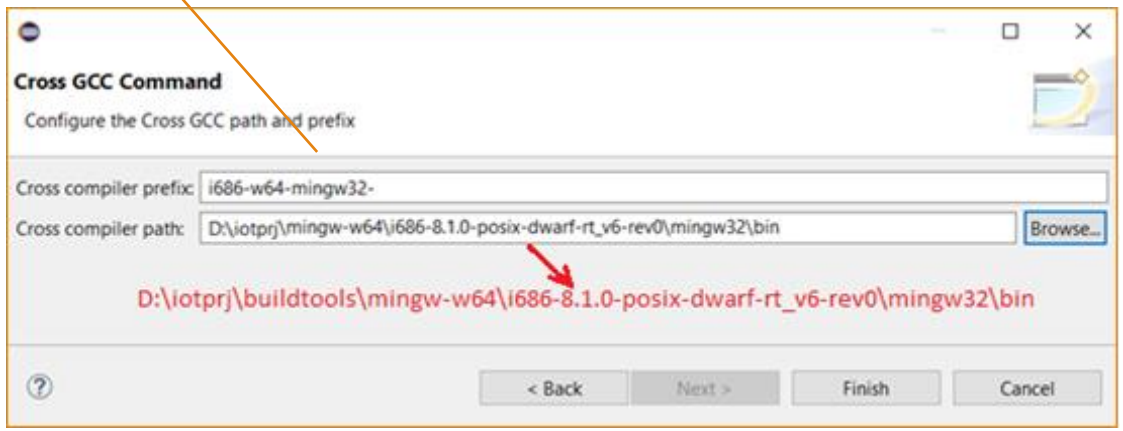


Eclipse C++ Project 3

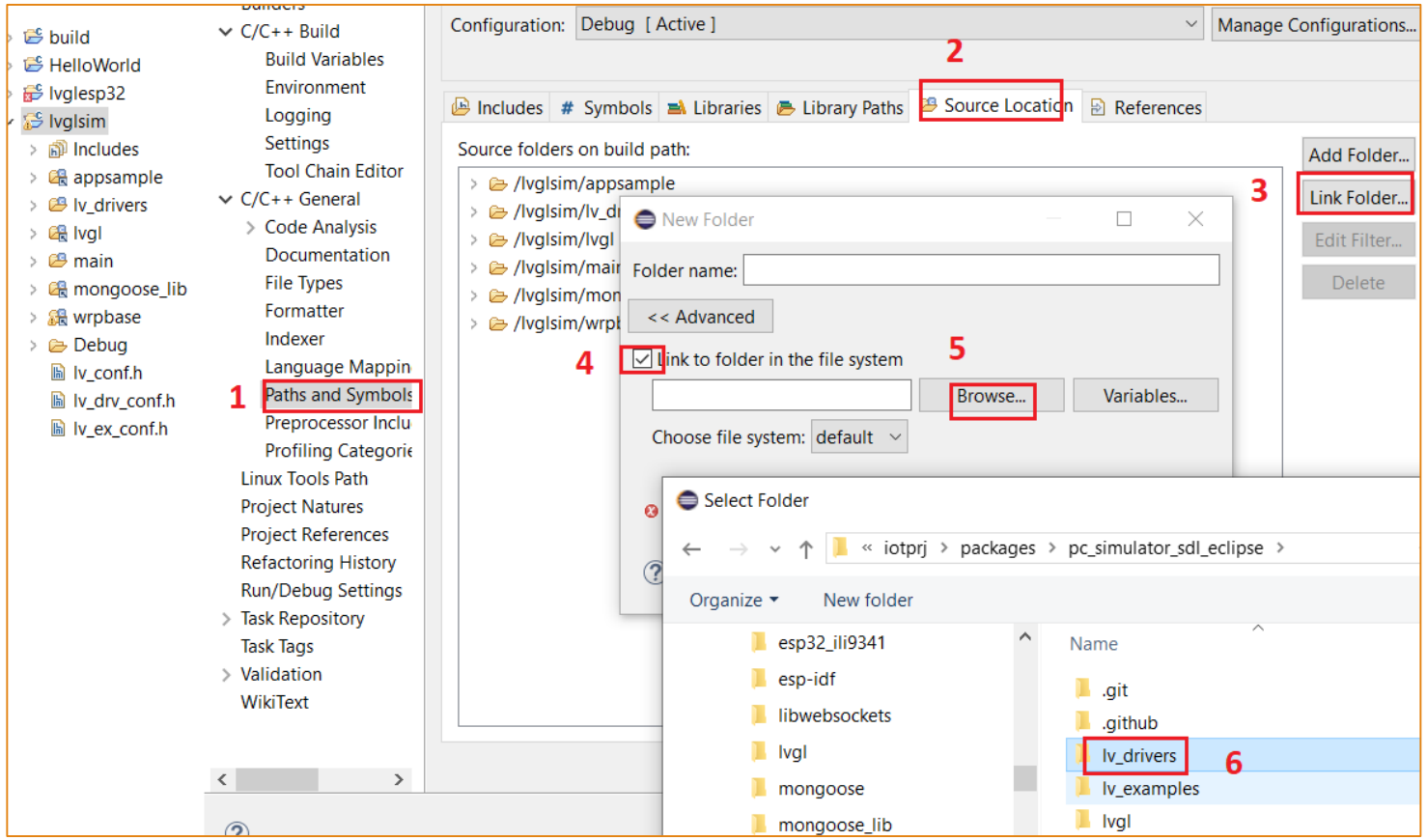




Simulator

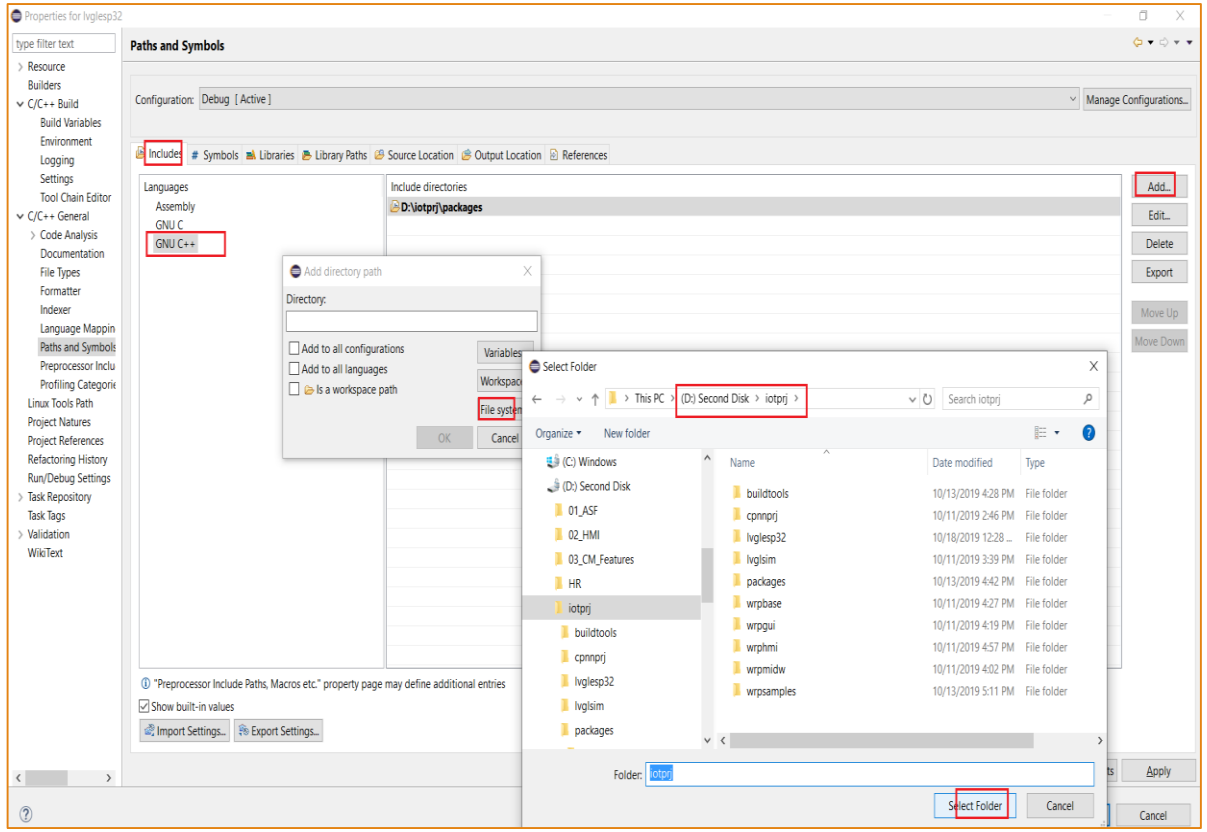


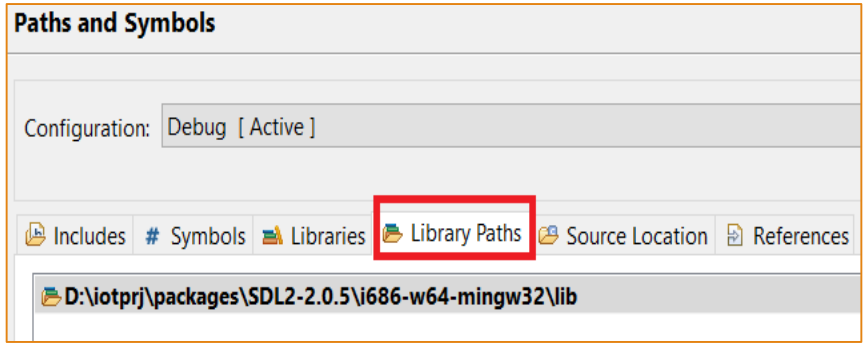
LVGL SIM Eclipse 1

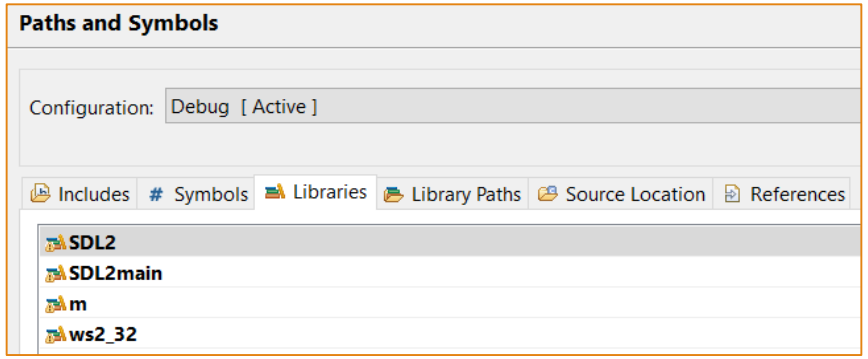


Configuration of Lvgl Simu Eclipse Project: Link source folders such as appsample, Lvgl, mongoose_lib, and wrpbase

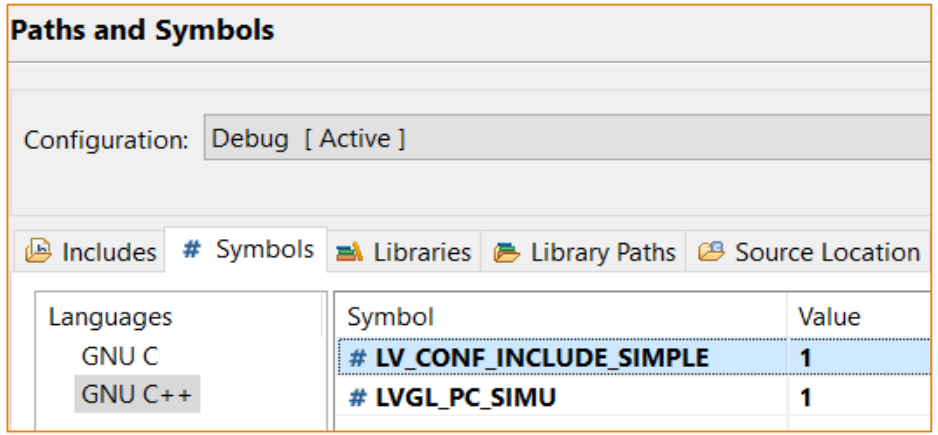
LVGL SIM Eclipse 2

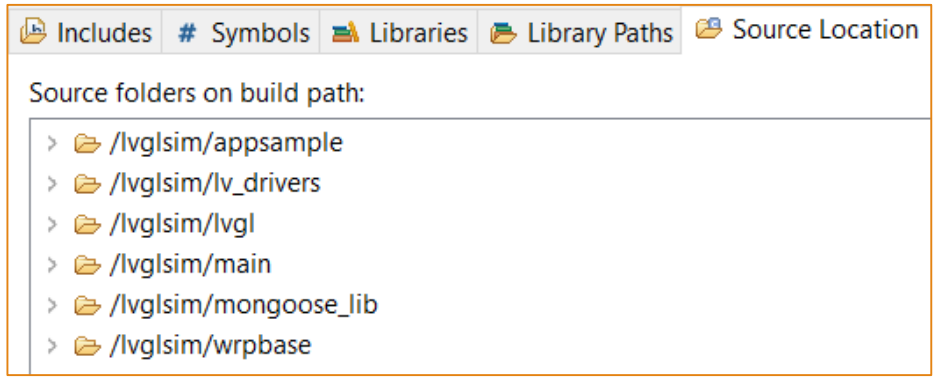


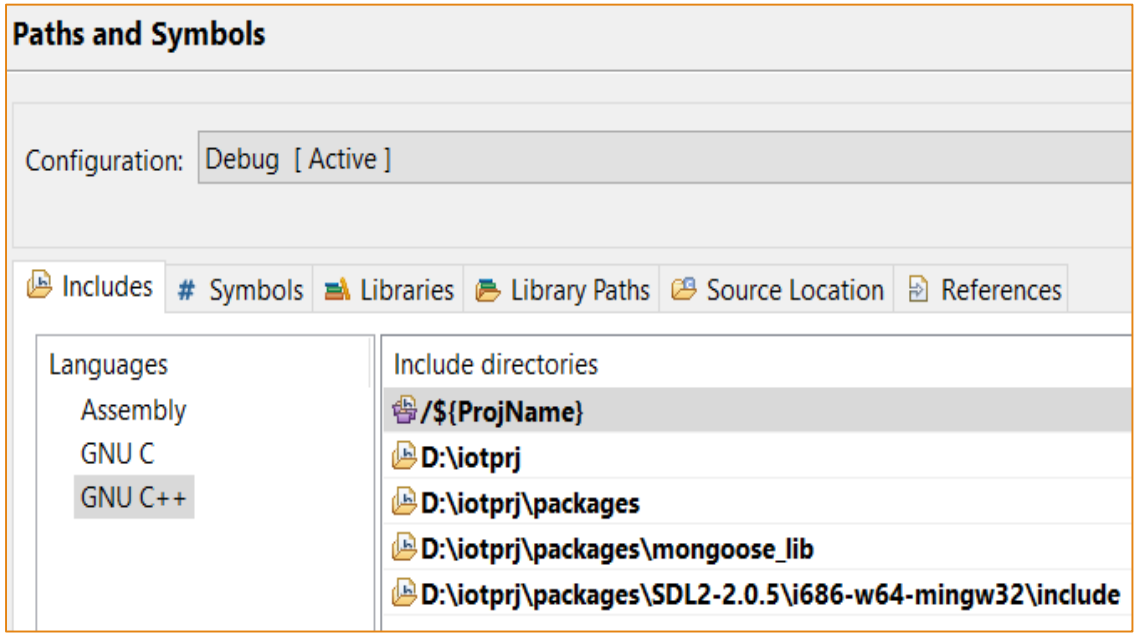




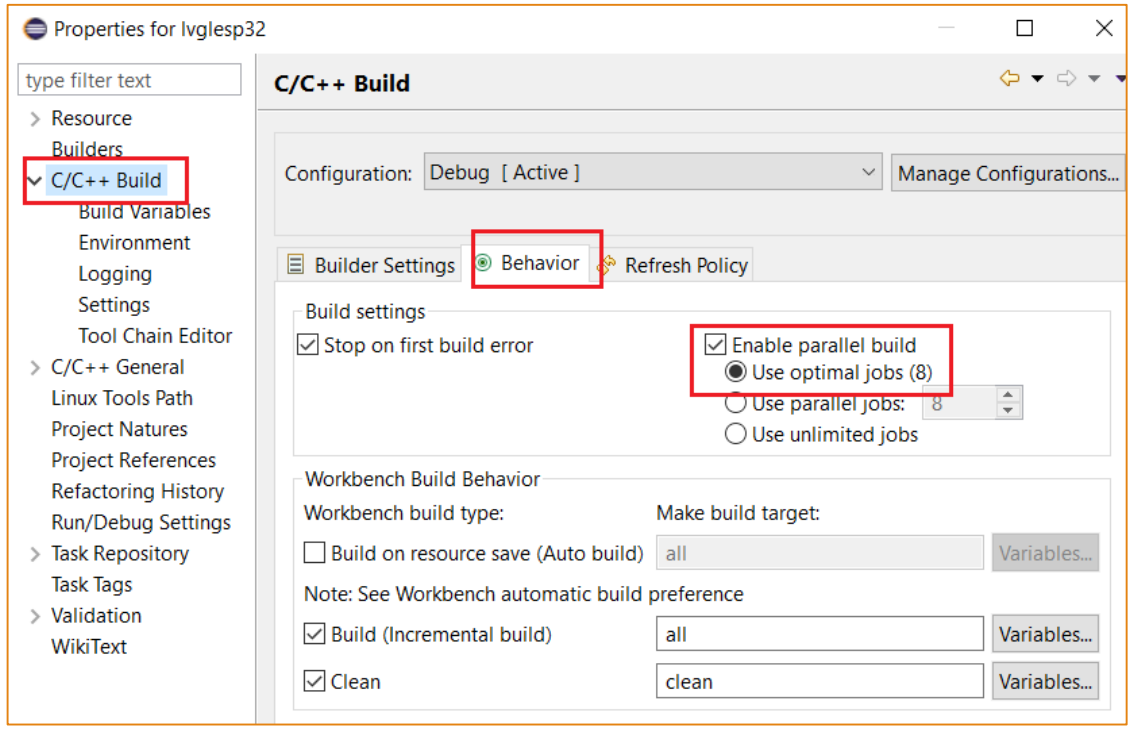
LVGL SIM Eclipse 3

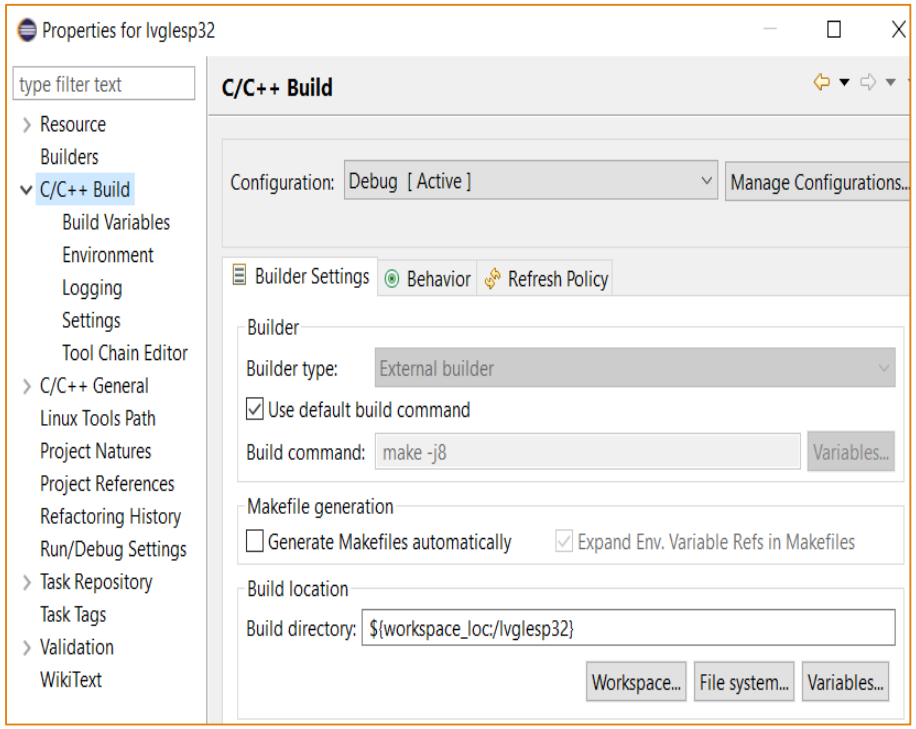






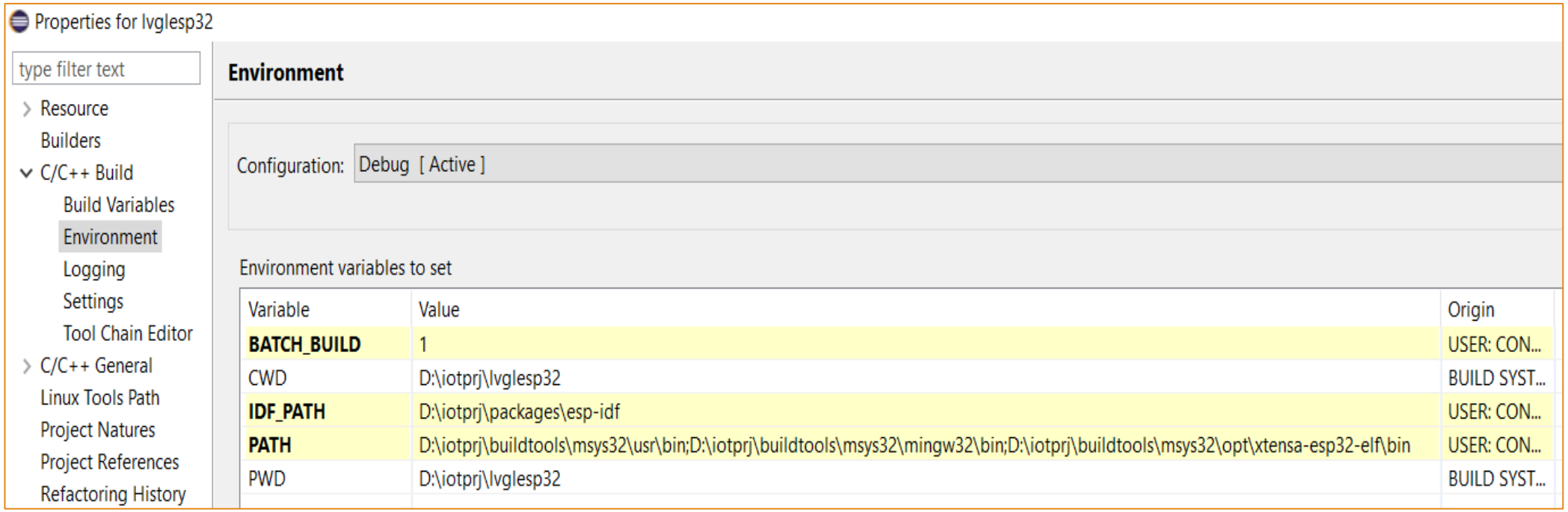
LVGL ESP32 Eclipse





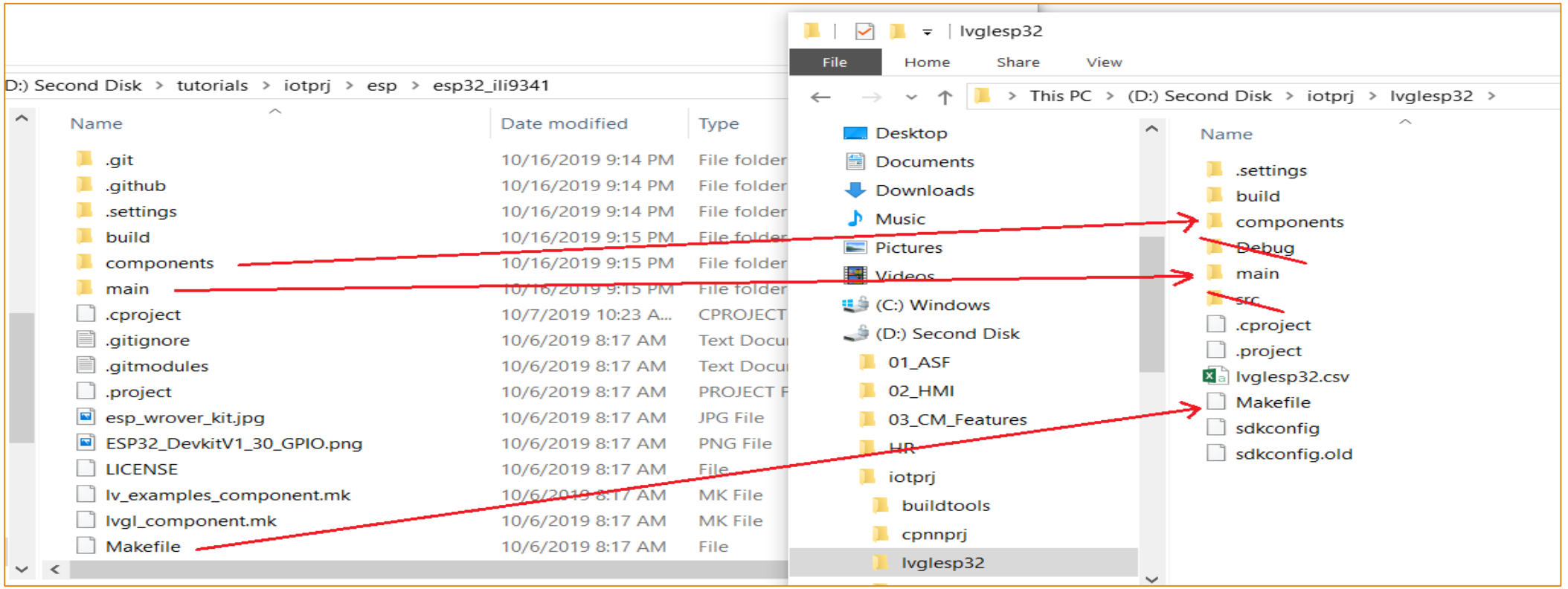
LVGL ESP32 Eclipse 1

Configure ESP-IDF framework and GCC for ESP32 compiler path as below

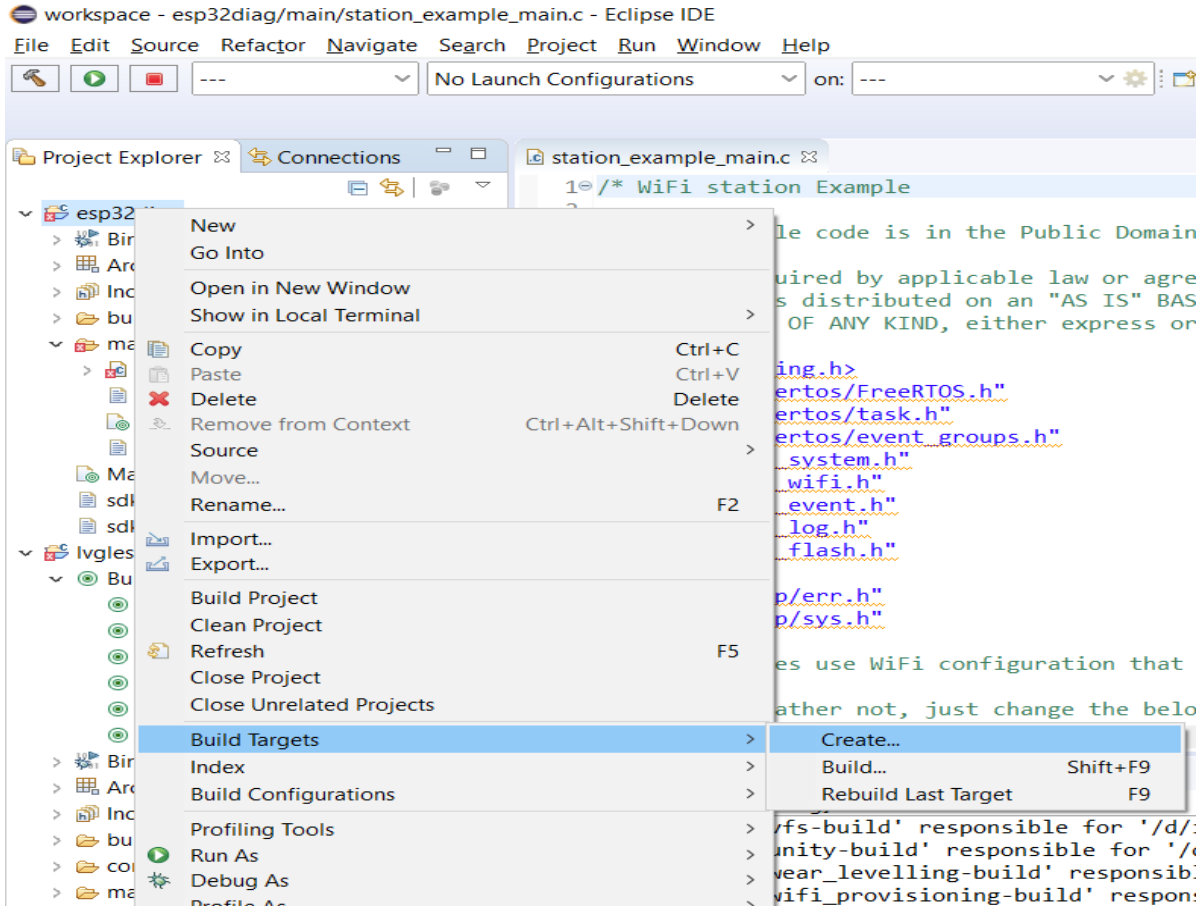


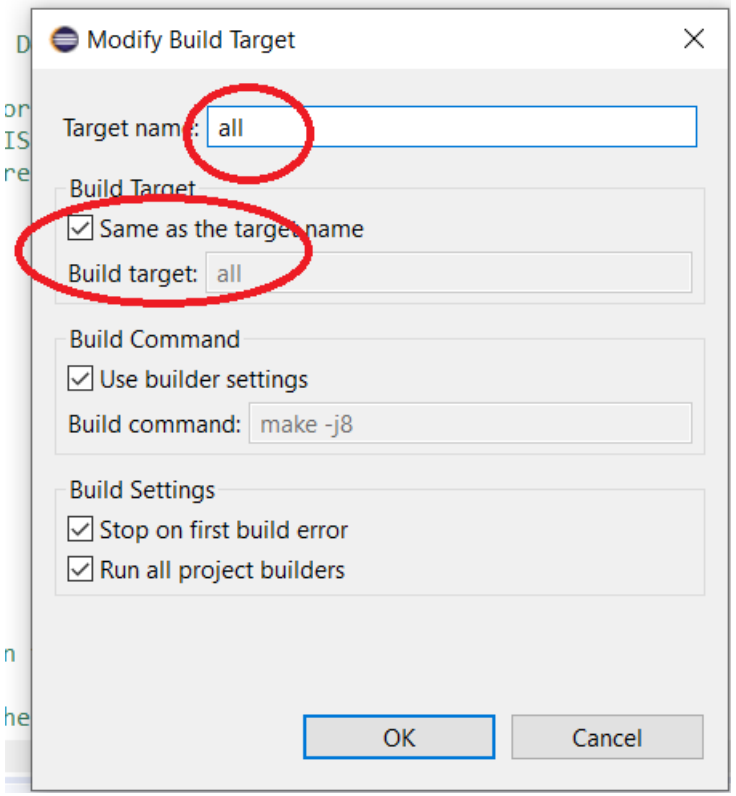
LVGL ESP32 Eclipse 2

Copy esp32_ili9341/**components**, **main** folder and **Makefile** to lvglesp32



LVGL ESP32 Eclipse 3



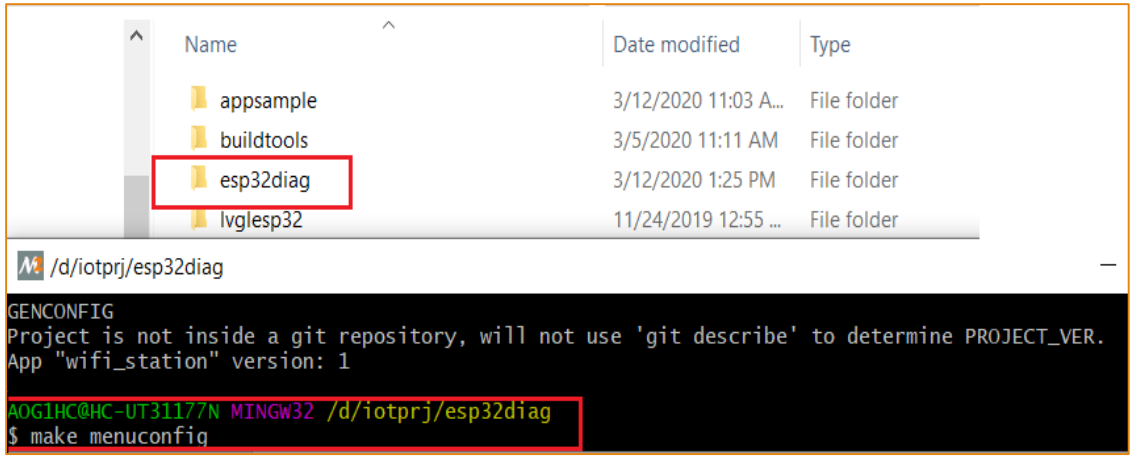


LVGL ESP32 Eclipse 4

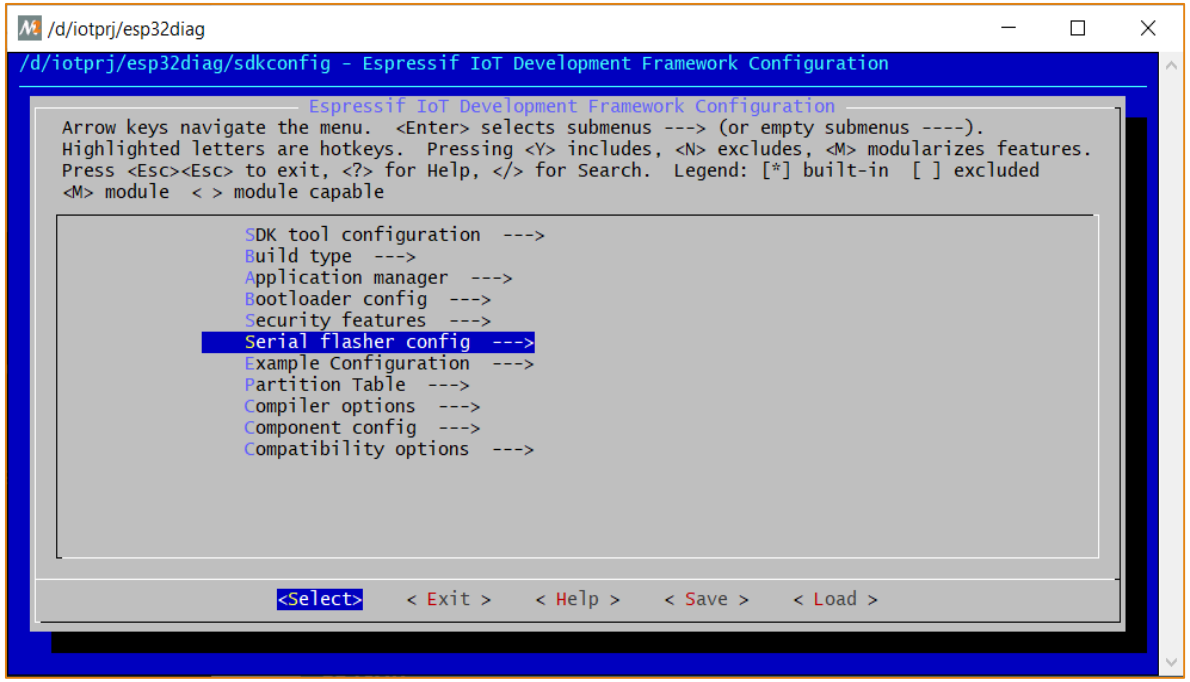
Then edit Makefile to connect to WrpBase framework as below:

```
# ESP32 Diagnosis Project Name
PROJECT_NAME := esp32diag
#For C++ language flags
EXTRA_CPPFLAGS := -DLV_CONF_INCLUDE_SIMPLE -DILI9341_BCKL_ACTIVE_LVL=0
#Use LVGL library and ILI9341 driver
EXTRA_COMPONENT_DIRS := D:/iotprj/packages/lvgl D:/iotprj/packages/esp32_ili9341/components/drv
#Include headers of lvgl component and headers of ili9341 driver
COMPONENT_EXTRA_INCLUDES := D:/iotprj/packages D:/iotprj/packages/esp32_ili9341/components
#Use include macros for wrpbase
CPPFLAGS += -DLVGL_PC_SIMU=0 -DLVGL_ESP32_ILI9341=1 -DUSE_ESP_IDF=1
#Use wrpbase component
EXTRA_COMPONENT_DIRS += D:/iotprj/wrpbase/wrpdrv D:/iotprj/wrpbase/wrpsys D:/iotprj/wrpbase/wrpgui
D:/iotprj/wrpbase/wrpmidw D:/iotprj/wrpbase/wrphmi D:/iotprj/wrpbase/wrpres
#esp-idf platform makefile
include $(IDF_PATH)/make/project.mk
```

Example of New ESP32 Project 1

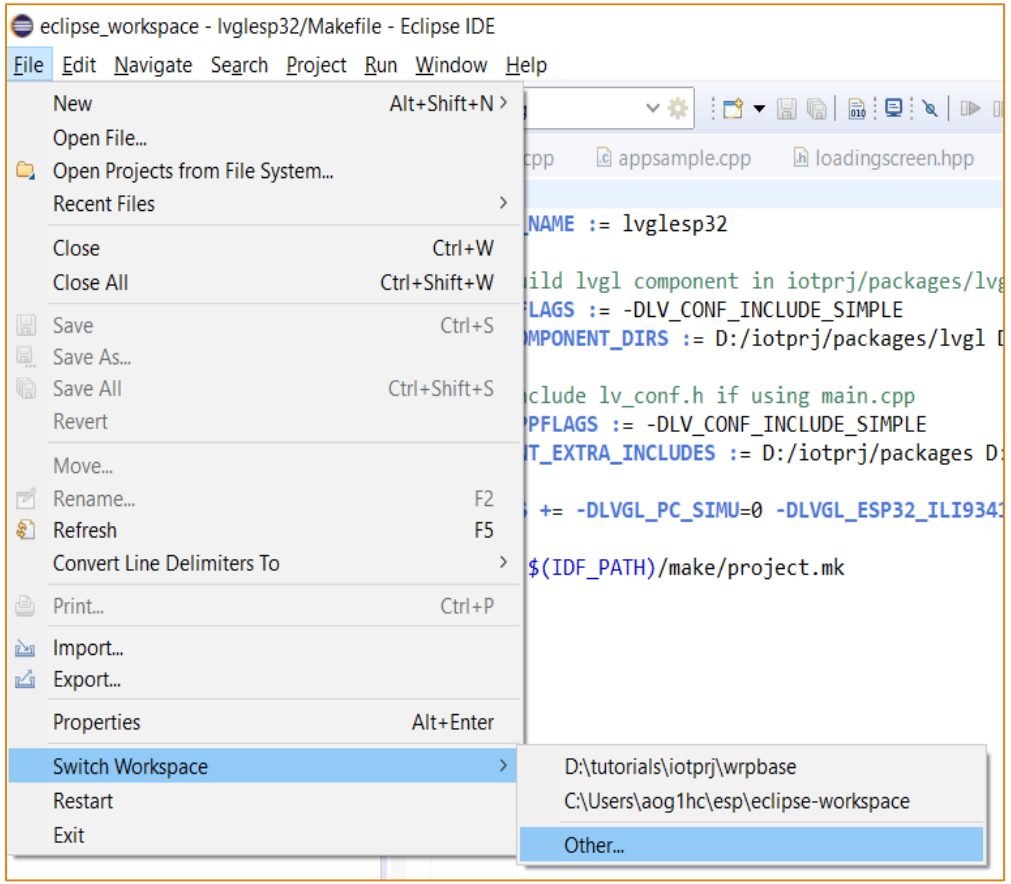


Example of the esp32diag project creation and WrpBase usage

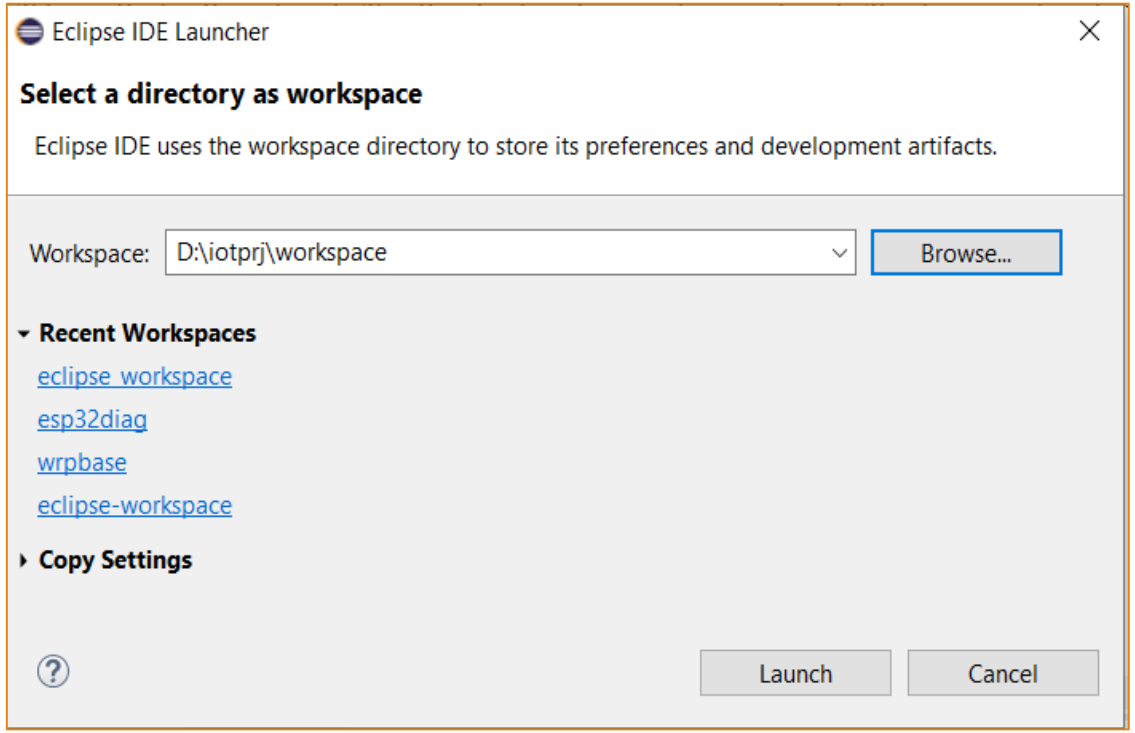


- Copy D:\iotprj\packages\esp-idf\examples\wifi\getting_started\station to D:\iotprj folder and rename to esp32diag
- Do “make menuconfig” and then “make”
- Now add it into Eclipse: create a workspace folder eg: D:\iotprj\workspace

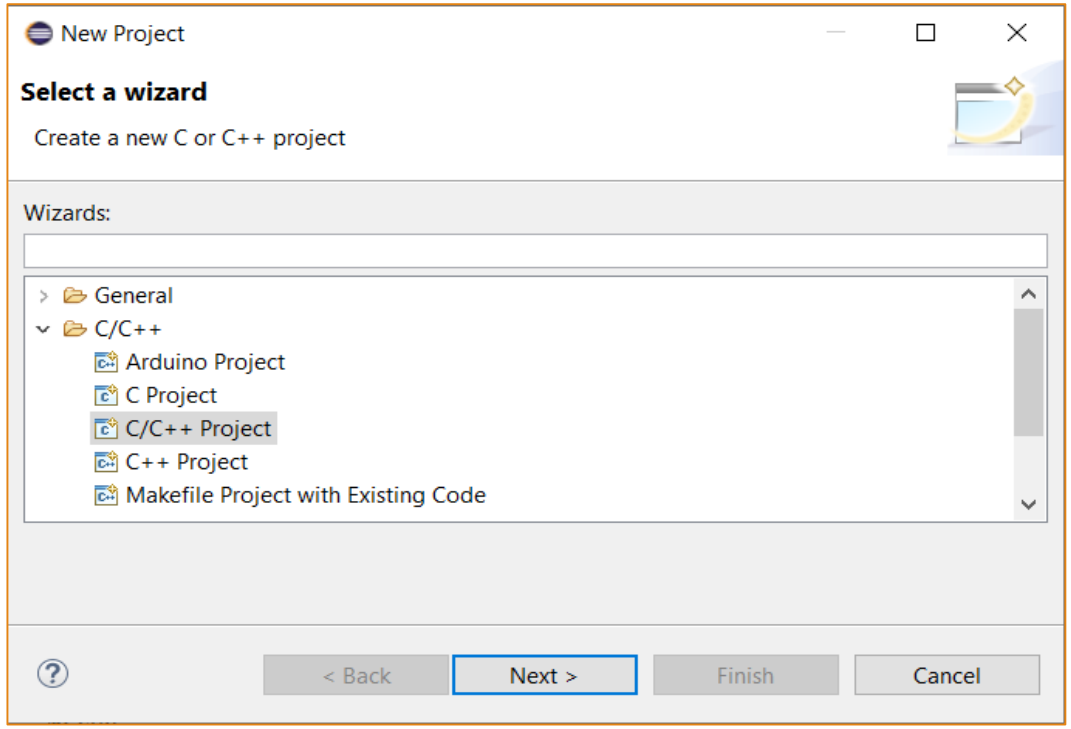
Example of New ESP32 Project 2

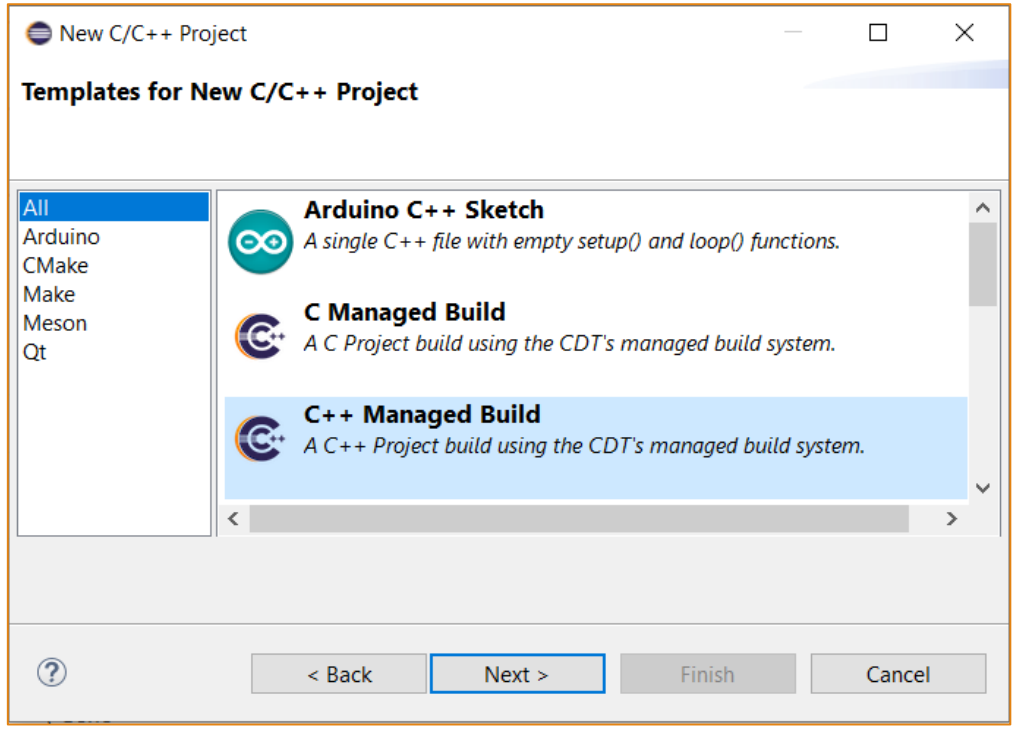


Open Eclipse and swith to D:\iotprj\workspace

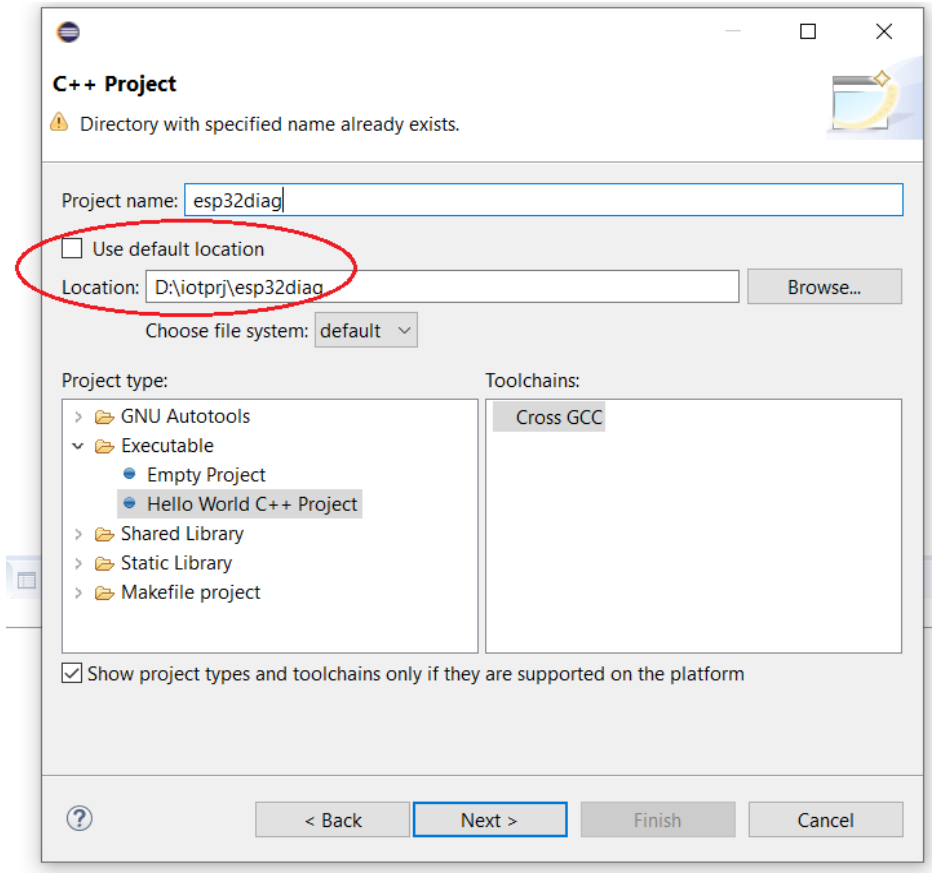


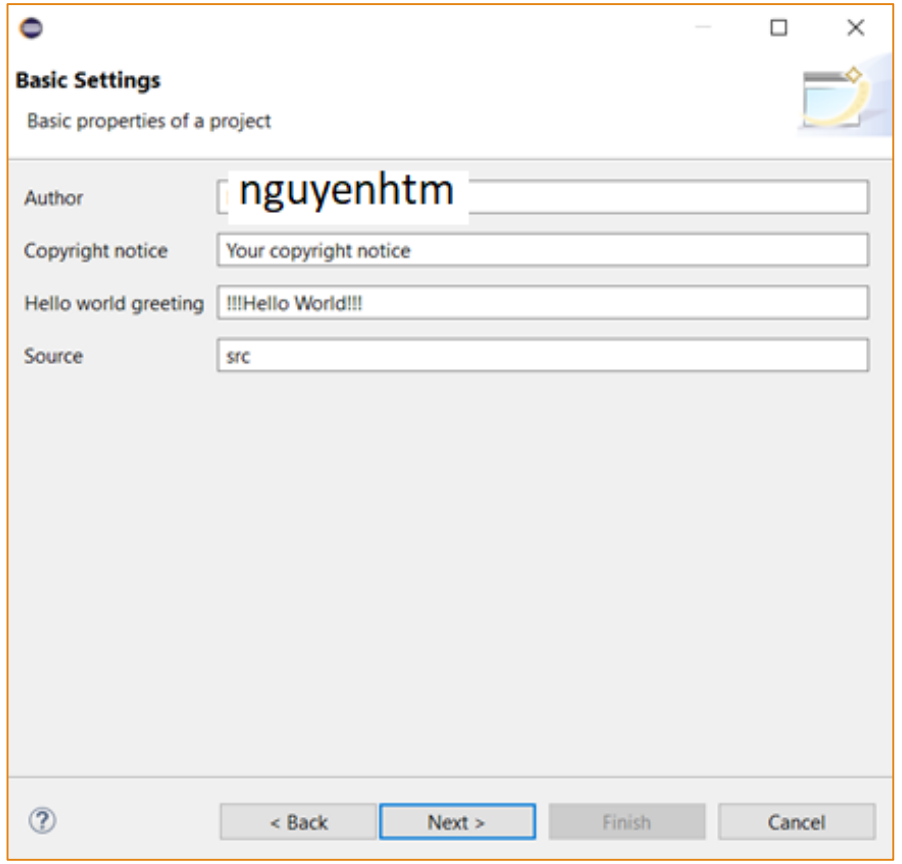
Example of New ESP32 Project 3



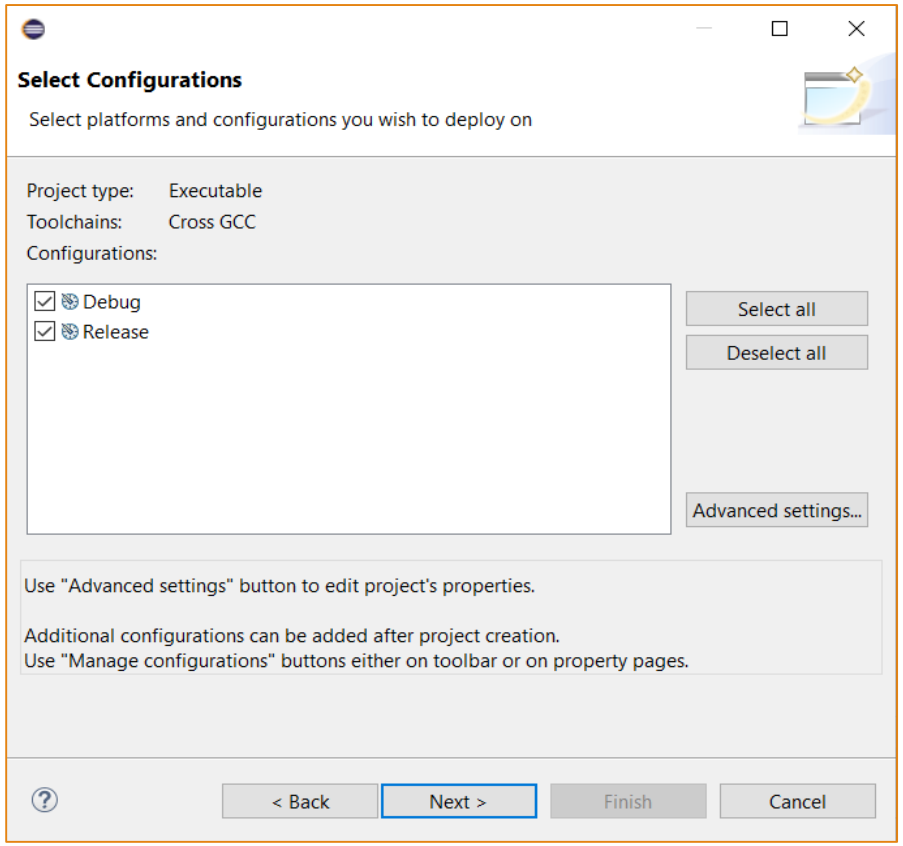


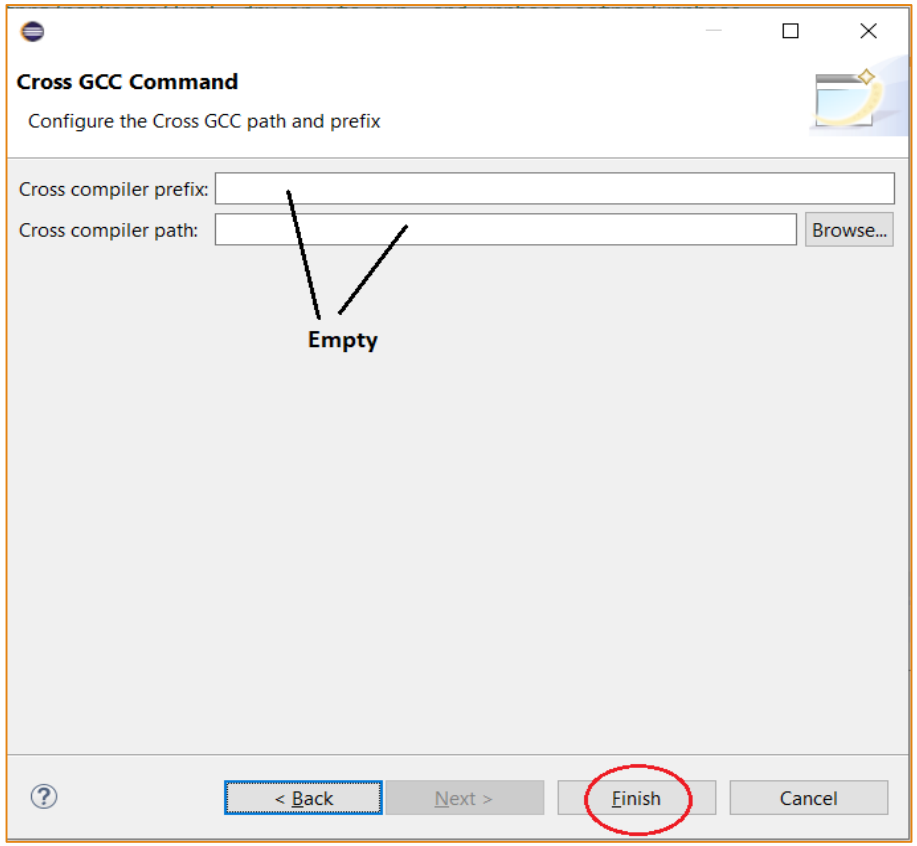
Example of New ESP32 Project 4



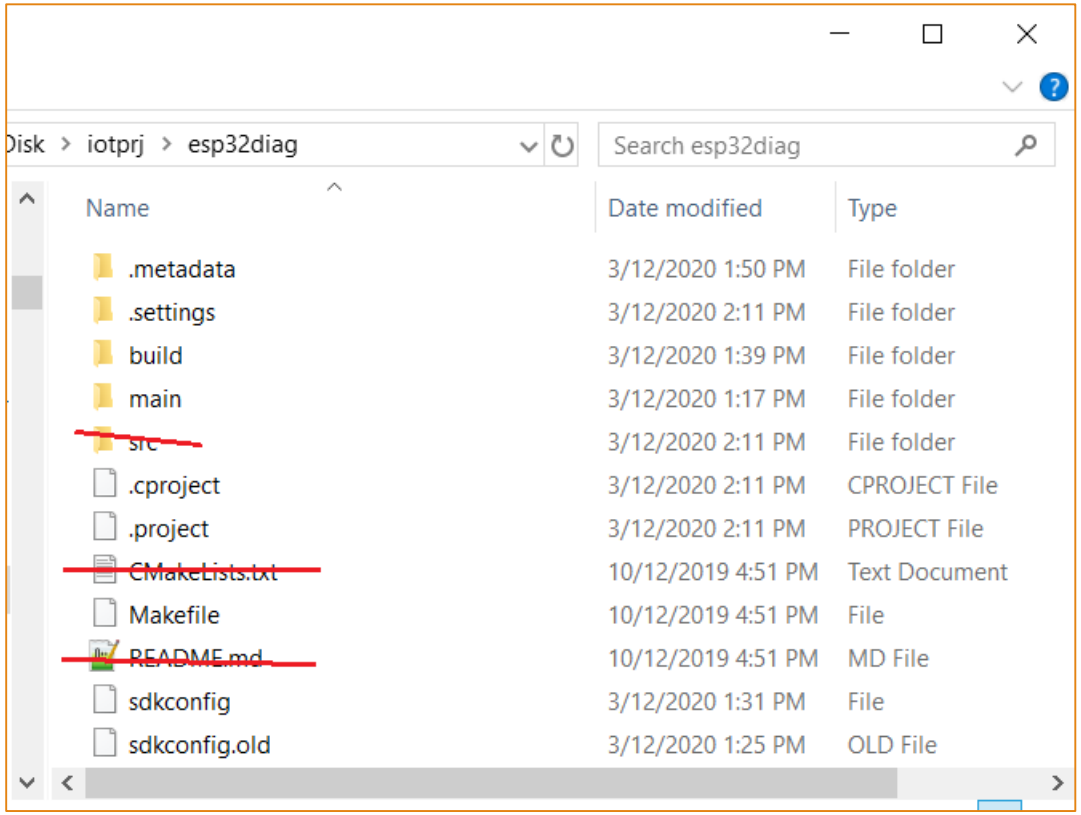


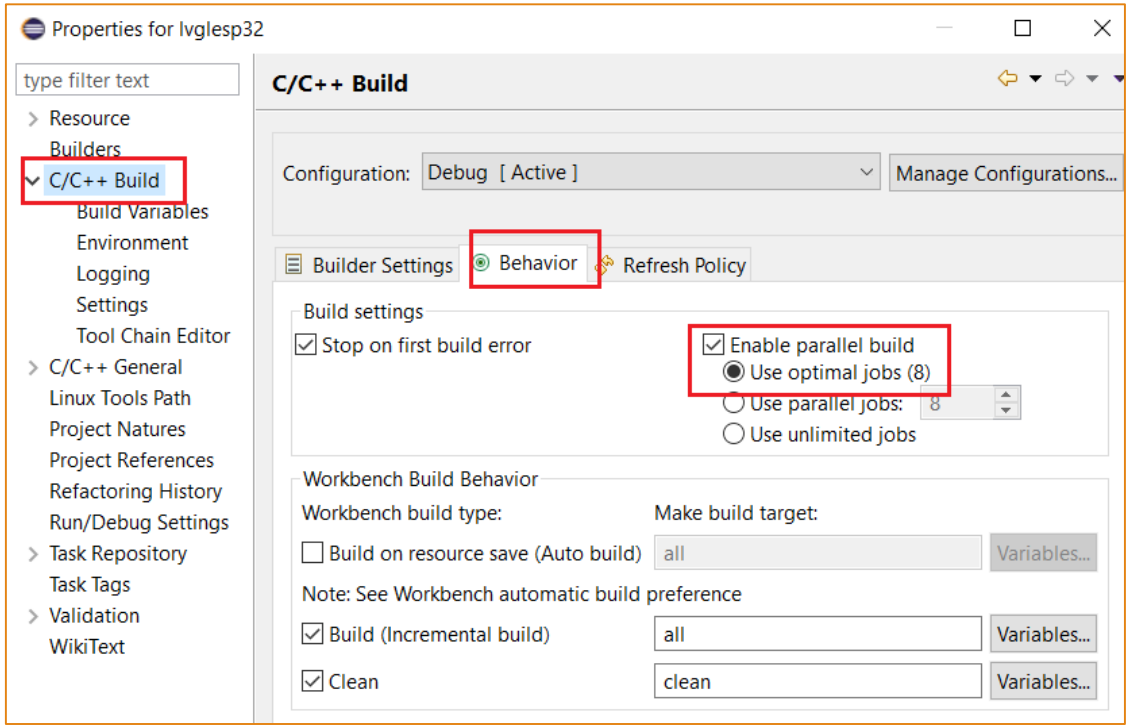
Example of New ESP32 Project 5



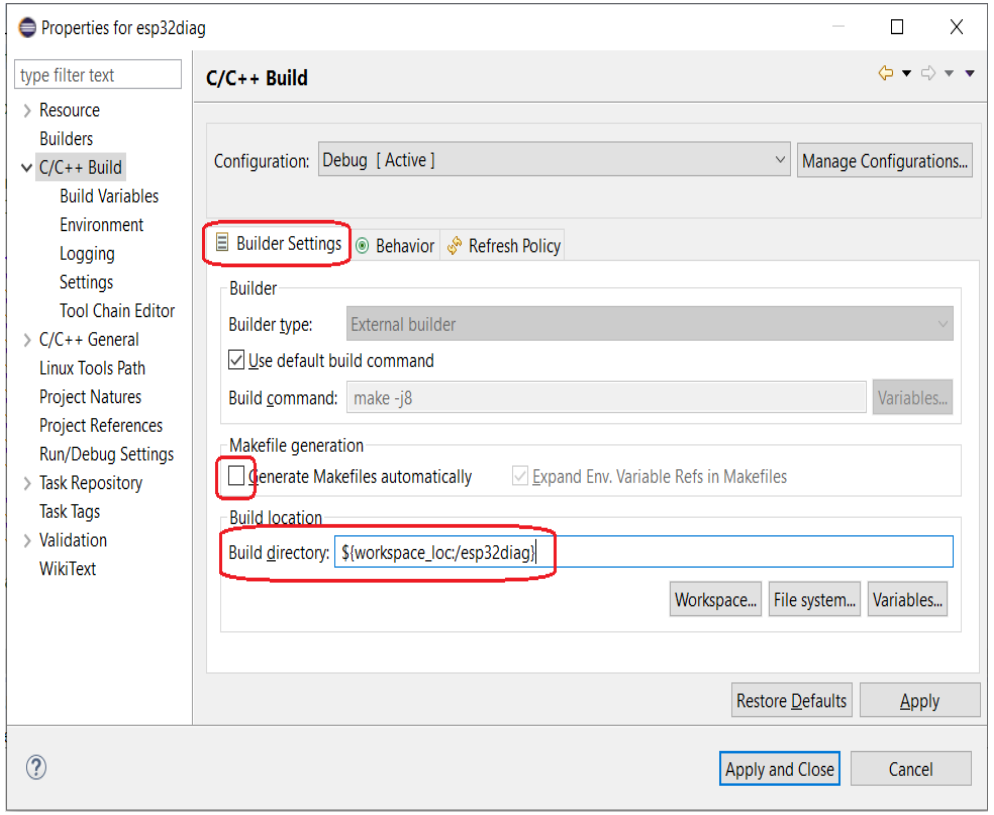


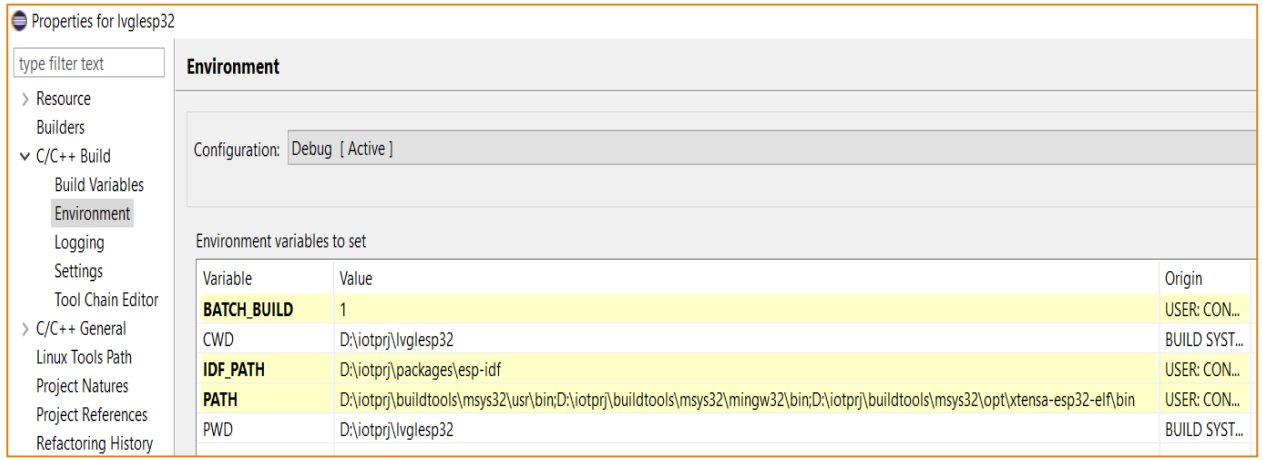
Example of New ESP32 Project 6



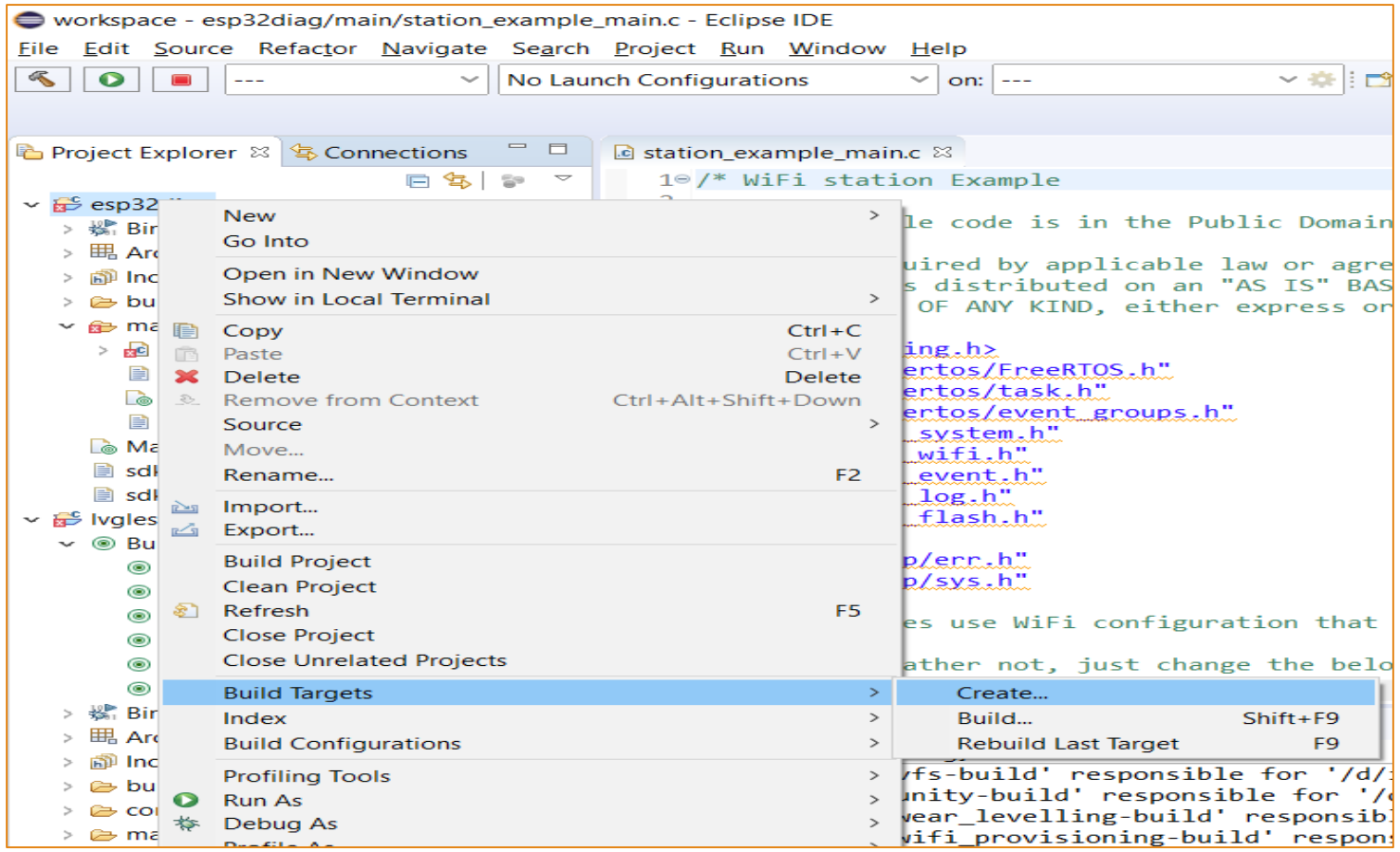


Example of New ESP32 Project 7



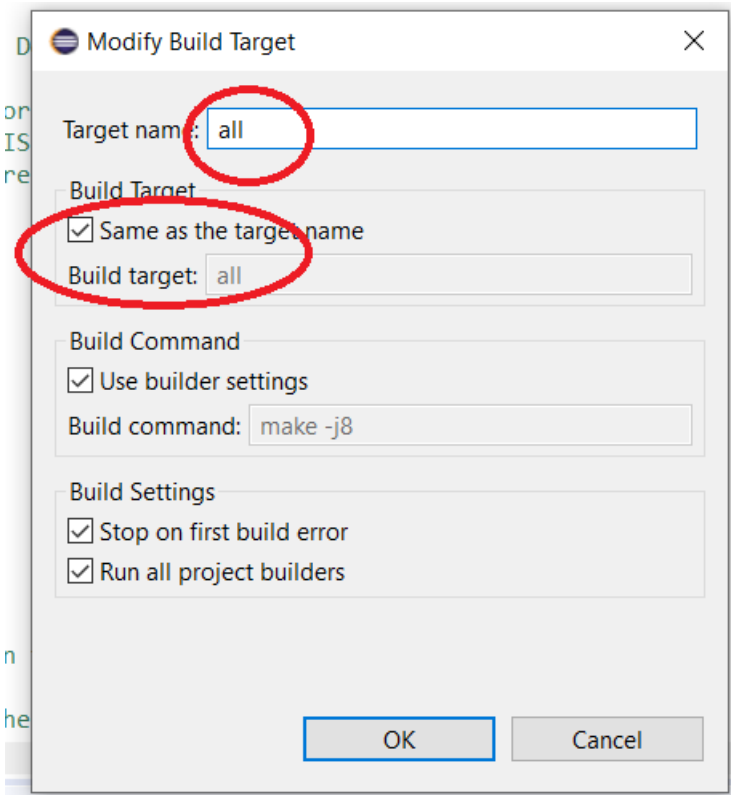


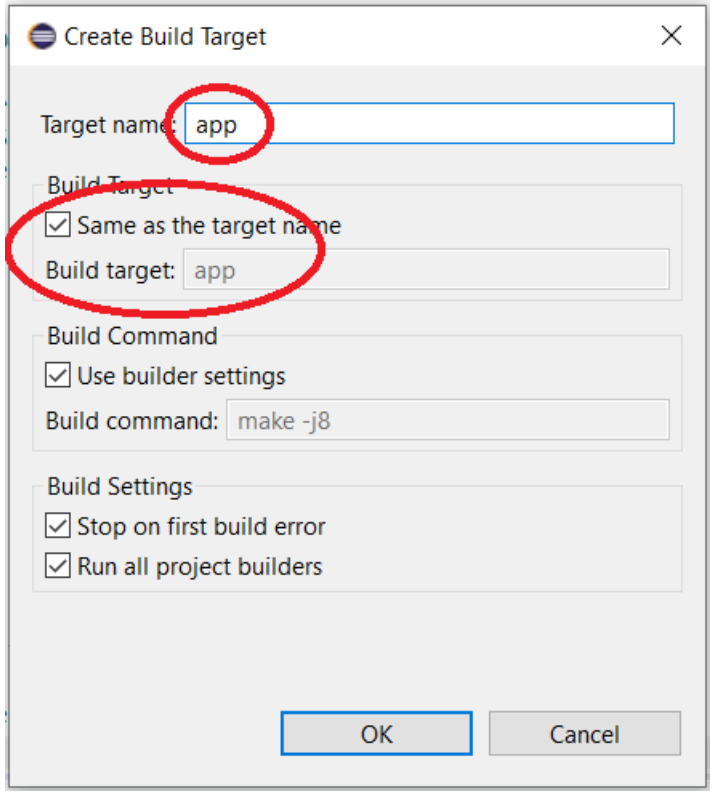
Example of New ESP32 Project 8



Create Build Target Icon

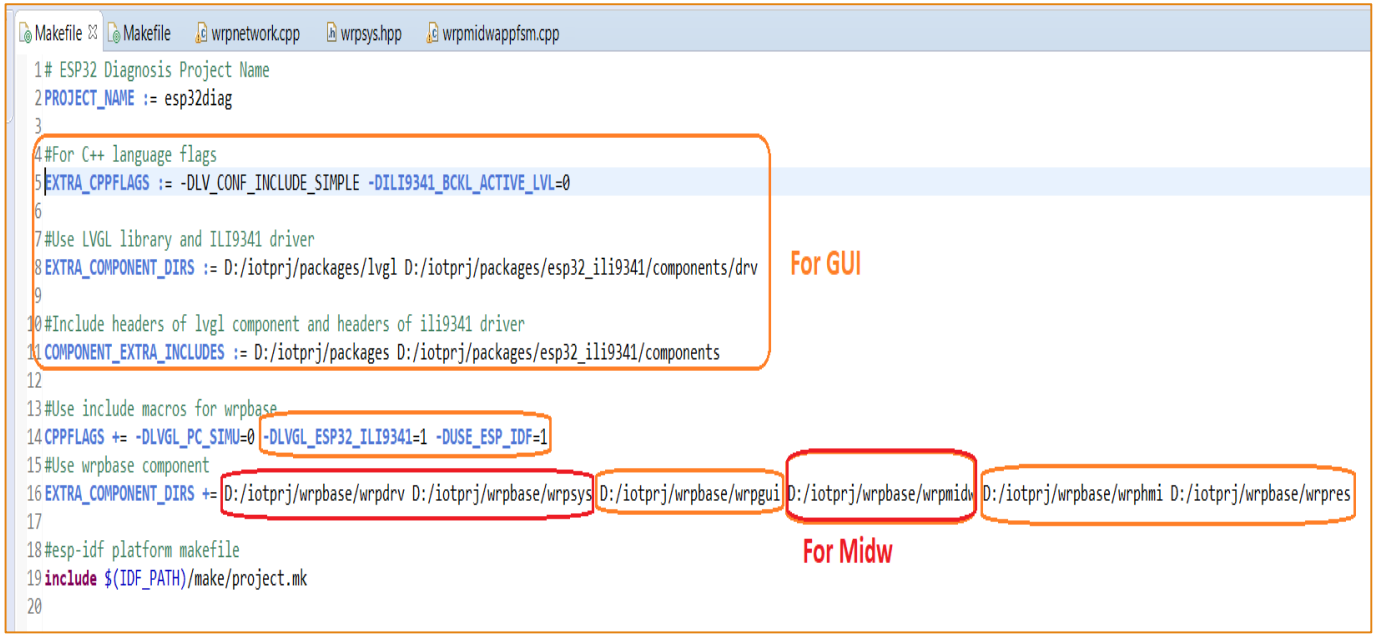
Example of New ESP32 Project 9





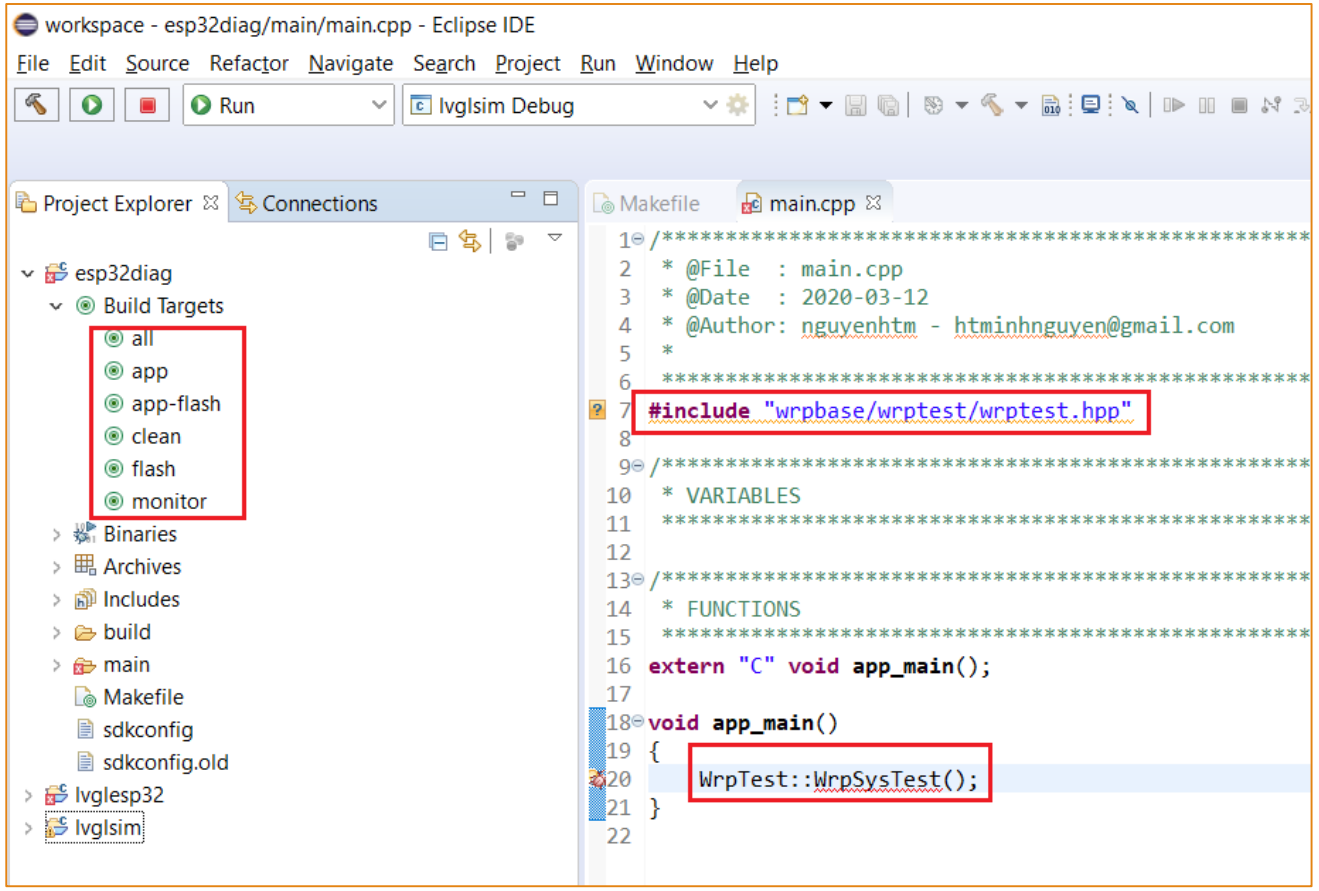
Similar for app-flash, clean, flash and monitor

Example of New ESP32 Project 10



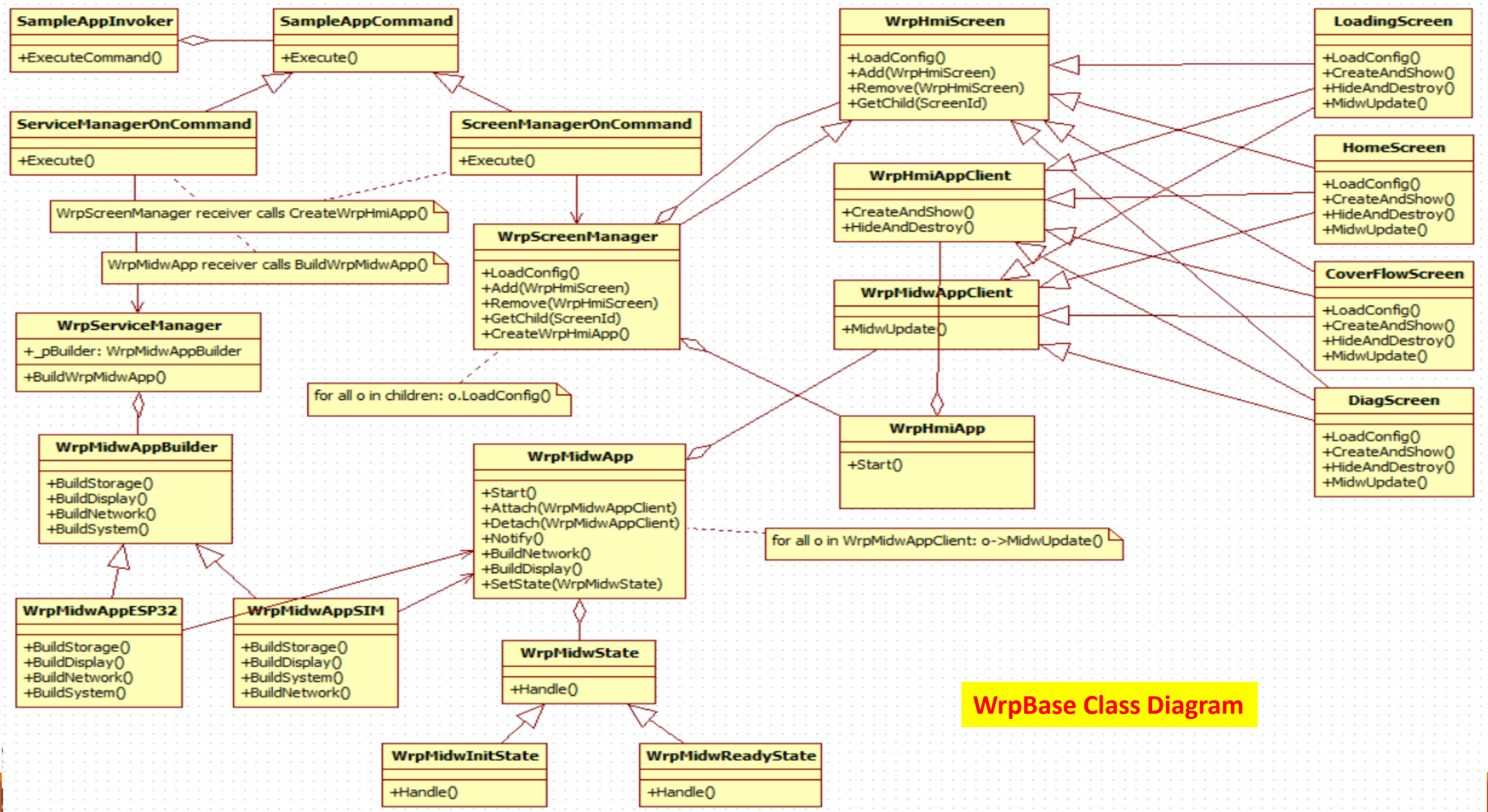
Edit Makefile to connect to WrpBase library

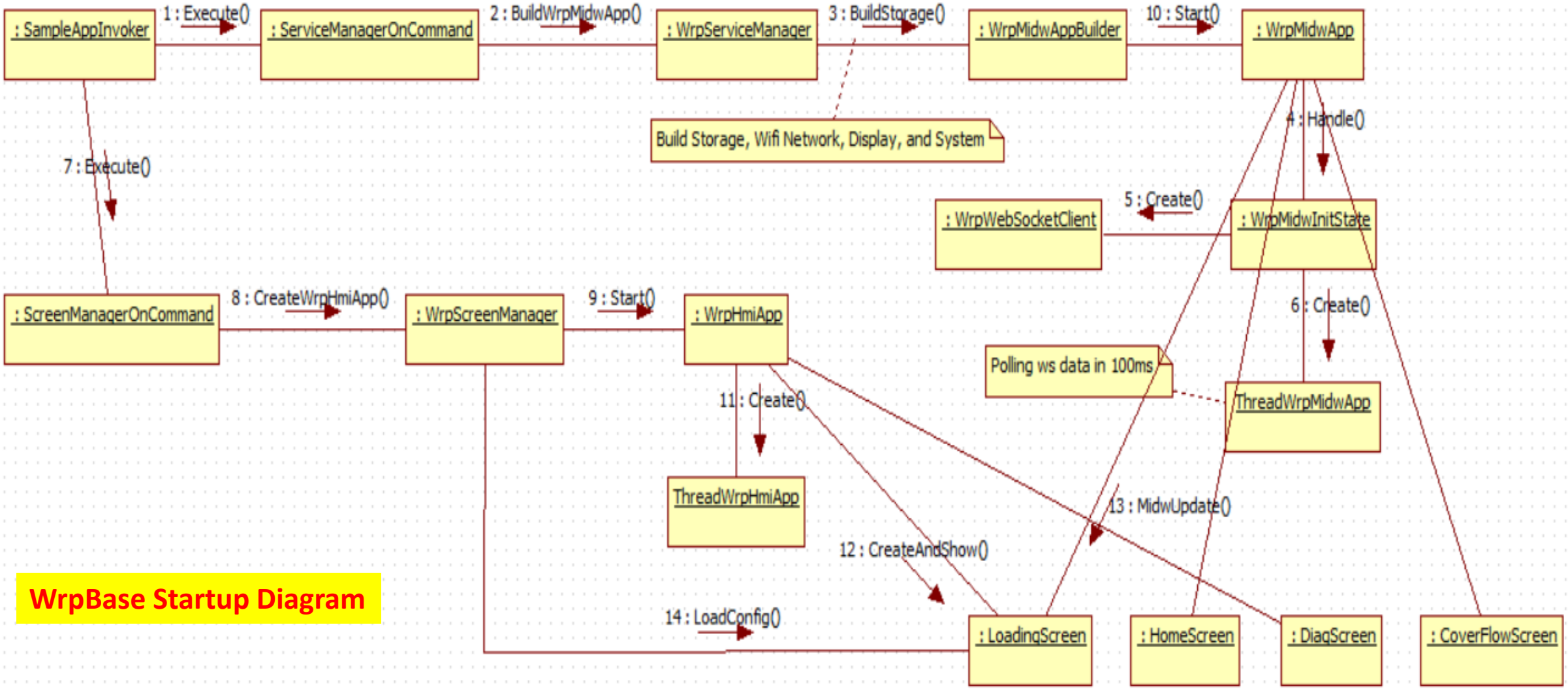
Example of New ESP32 Project 11



- Edit main.cpp to execute the system test function
- Click Build Targets -> app-flash to build and download into ESP32 target

DONE

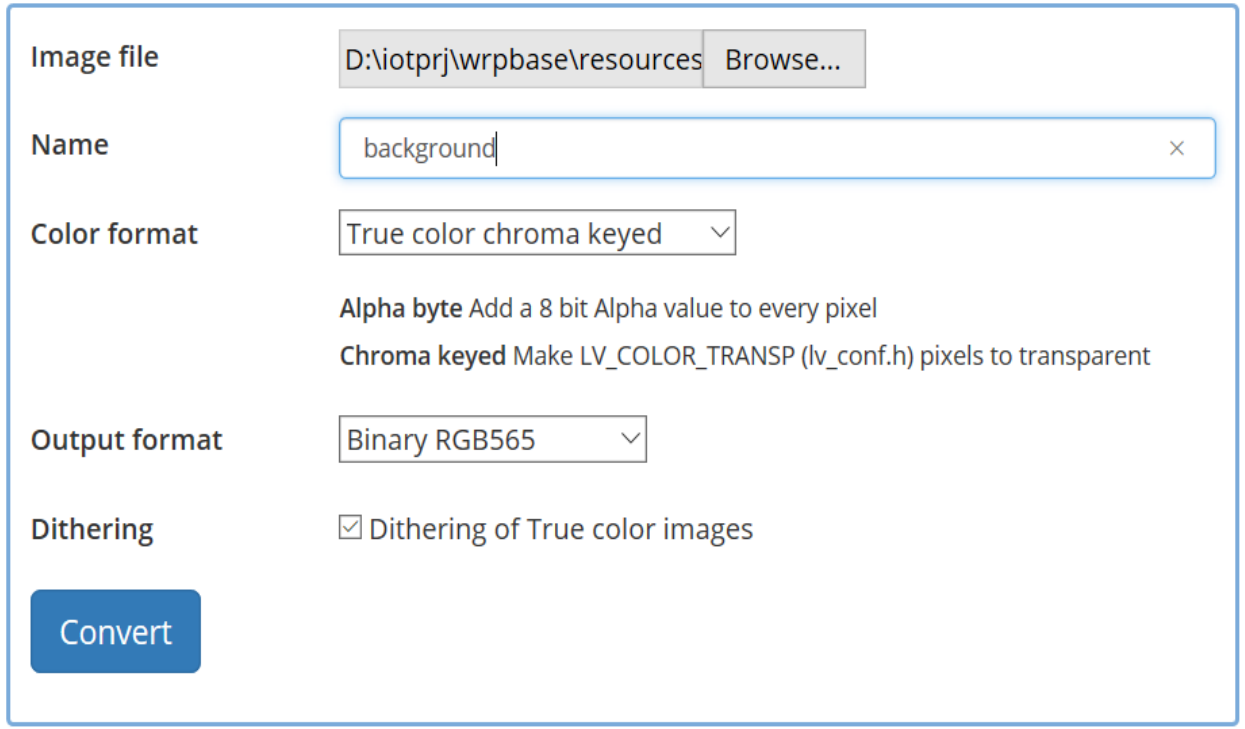

SampleAppInvoker
+ExecuteCommand()
SampleAppCommand
+Execute()
ServiceManagerOnCommand
+Execute()
ScreenManagerOnCommand
+Execute()
WrpScreenManager receiver calls CreateWrpHmiApp()
WrpMidwApp receiver calls BuildWrpMidwApp()
WrpServiceManager
+_pBuilder: WrpMidwAppBuilder
+BuildWrpMidwApp()
WrpScreenManager
+LoadConfig()
+Add(WrpHmiScreen)
+Remove(WrpHmiScreen)
+GetChild(ScreenId)
+CreateWrpHmiApp()
for all o in children: o.LoadConfig()
WrpHmiScreen
+LoadConfig()
+Add(WrpHmiScreen)
+Remove(WrpHmiScreen)
+GetChild(ScreenId)
WrpHmiAppClient
+CreateAndShow()
+HideAndDestroy()
WrpMidwAppClient
+MidwUpdate()
WrpHmiApp
+Start()
LoadingScreen
+LoadConfig()
+CreateAndShow()
+HideAndDestroy()
+MidwUpdate()
HomeScreen
+LoadConfig()
+CreateAndShow()
+HideAndDestroy()
+MidwUpdate()
CoverFlowScreen
+LoadConfig()
+CreateAndShow()
+HideAndDestroy()
+MidwUpdate()
DiagScreen
+LoadConfig()
+CreateAndShow()
+HideAndDestroy()
+MidwUpdate()
WrpMidwAppBuilder
+BuildStorage()
+BuildDisplay()
+BuildNetwork()
+BuildSystem()
WrpMidwApp
+Start()
+Attach(WrpMidwAppClient)
+Detach(WrpMidwAppClient)
+Notify()
+BuildNetwork()
+BuildDisplay()
+SetState(WrpMidwState)
for all o in WrpMidwAppClient: o->MidwUpdate()
WrpMidwAppESP32
+BuildStorage()
+BuildDisplay()
+BuildNetwork()
+BuildSystem()
WrpMidwAppSIM
+BuildStorage()
+BuildDisplay()
+BuildSystem()
+BuildNetwork()
WrpMidwState
+Handle()
WrpMidwInitState
+Handle()
WrpMidwReadyState
+Handle()
WrpBase Class Diagram



WrpBase Startup Diagram

Image Converter

- https://littlevgl.com/image-converter: the current configuration for image converter



References

LVGL: https://github.com/littlevgl/lvgl

ESP-IDF: https://docs.espressif.com/projects/esp-idf/en/latest/index.html

Patterns: Command, Singleton, Observer, Factory Method, Builder, State, Composite