ARM/JLink Setup

Author: Dipsy Wong (dipsywong98)

This section will cover how to set up eclipse for flashing program to MCU and setting up the development environment, which will take you around 1hour including downloading software. If you are unlucky, it may even take you a whole day.

Video walkthrough: https://youtu.be/v834HF_Uha4

If you want to set up eclipse for console program, please refer to https://github.com/mcreng/SmartCarSWTutorial18/blob/master/0b_mingw_setup.md

1.Download and Install the Essential Software

- Eclipse C++(Suggested version: Neon1, never use Neon2)

  direct download links:

  - Windows 32bit
  - Windows 64bit
  - MacOS
  - Linux 32 bit
  - Linux 64 bit

- GNU Tools for ARM (Suggested version: 4.8 2014q3, which is not the newest)

  direct download links:

  - Windows
  - Linux
  - MacOS

- Segger J-Link software pack (Suggested version V614b, which is not the newest also)

  direct download links:

  - Windows
  - MacOS
  - Linux 32bit
  - Linux 64bit

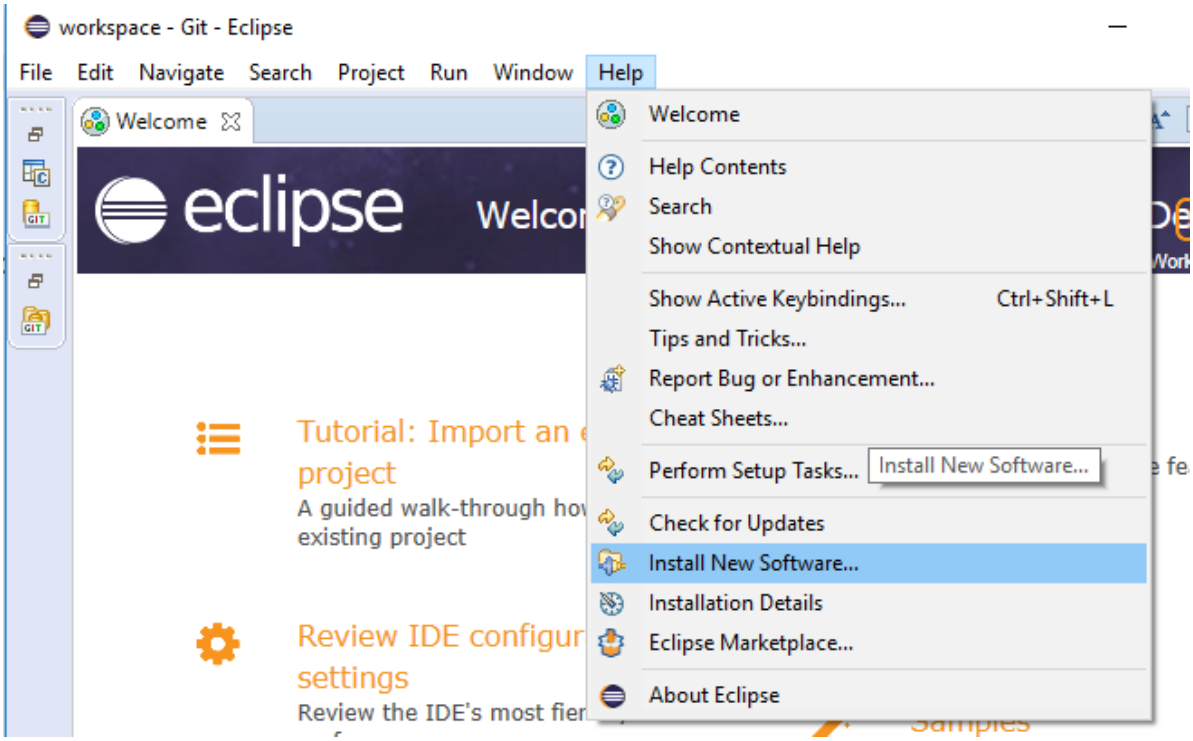
- GNU Make

  - Windows
  - Mac (if that doesn't work, find another one at https://code.google.com/archive/p/rudix/downloads)

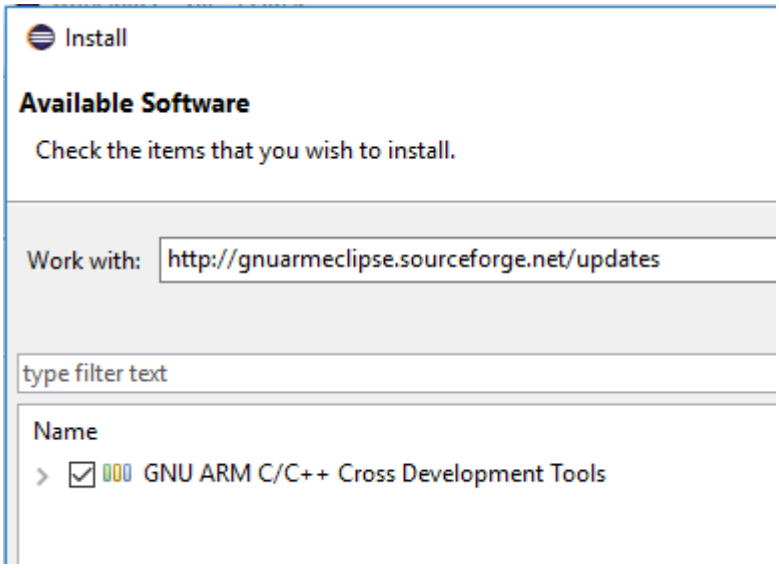
2. Install Plug-in for Eclipse

https://youtu.be/v834HF_Uha4?t=35s

- GNU ARM eclipse Plugins
  - [Help]->[Install New Software...]



- Type http://gnuarmeclipse.sourceforge.net/updates in the textbox next to [Work with] label
  - Press Enter
  - Check the item [GNU ARM C/C++] in the CheckedListBox below



  - Press [Next]->[Next]->Accept license agreement & [finish]

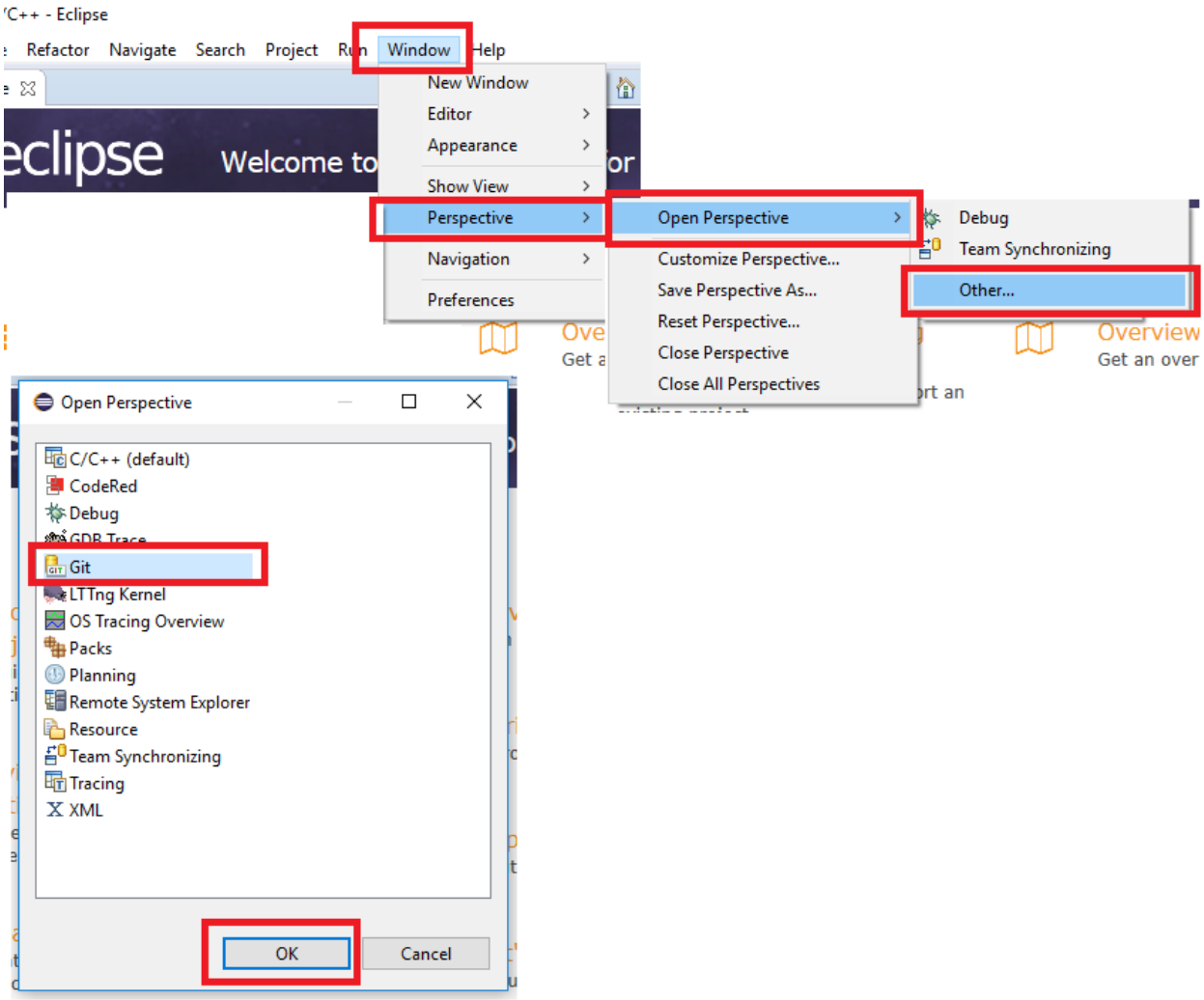
- EGit
  - [Help]->[Install New Software...]
  - Type http://download.eclipse.org/egit/updates in the textbox next to [Workwith] label
    - Press Enter
    - Check the items [Git integration for Eclipse] & [Java implementation of Git] in the CheckedListBox below
    - Press [Next]->[Next]->Accept license agreement & [finish]

3. Library Setting

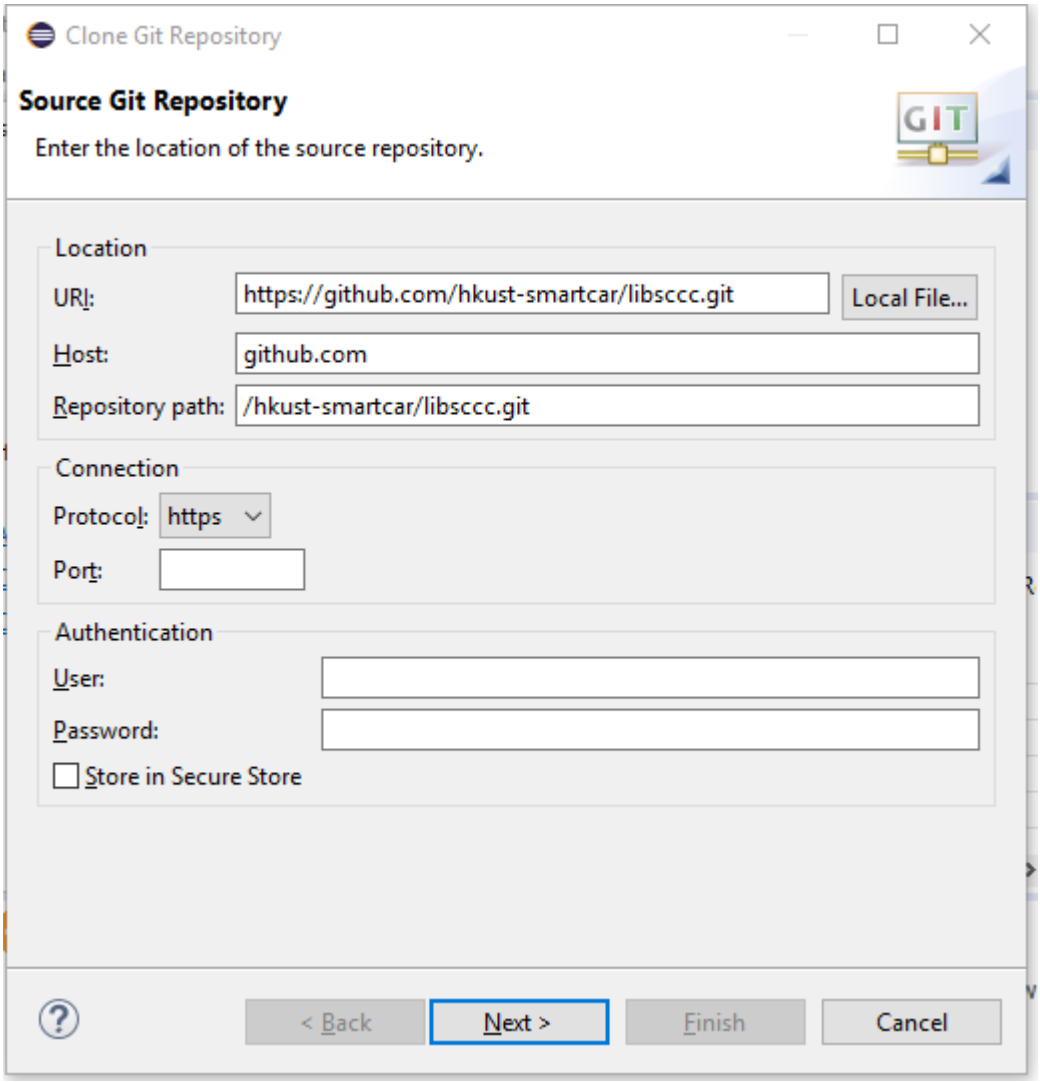
https://youtu.be/v834HF_Uha4?t=2m29s

3.1 Download the Library by Git Pull

- [Window]->[Open Perspective]->[Other...]->Select [Git]->[OK]



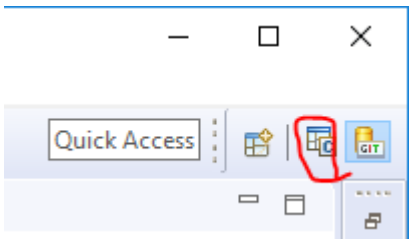
- Copy https://github.com/hkust-smartcar/libsccc.git (Library)
  - Click [Git Repositories] tab
  - { Ctrl + V } / RightClick & [Paste Repository Path or URI]
  - [Next] -> [Next] -> [Finish]



- Right Click [libsccc] -> [Pull]

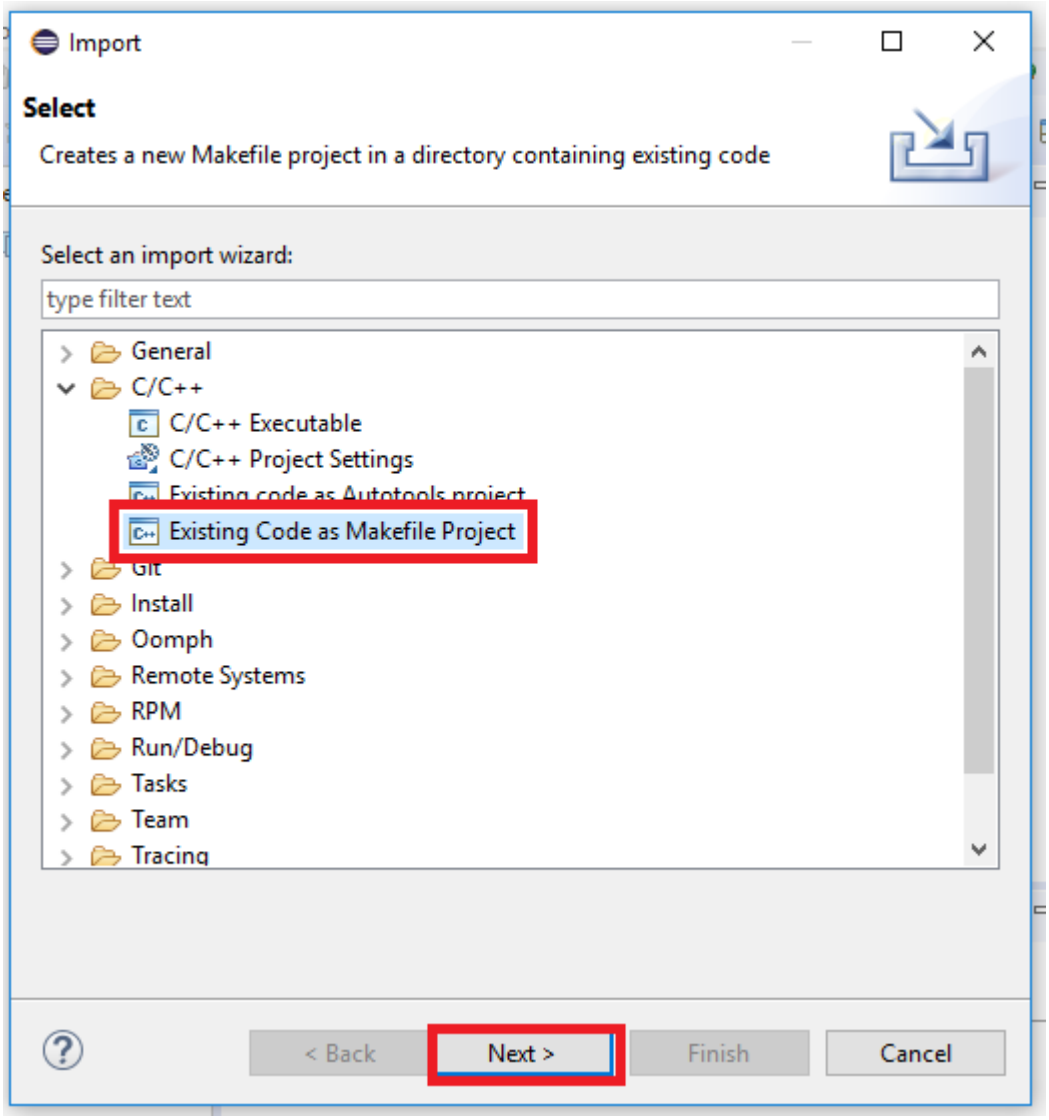
3.2 Import the Library to Eclipse

- Goto [C/C++ perspective] on the top-right corner

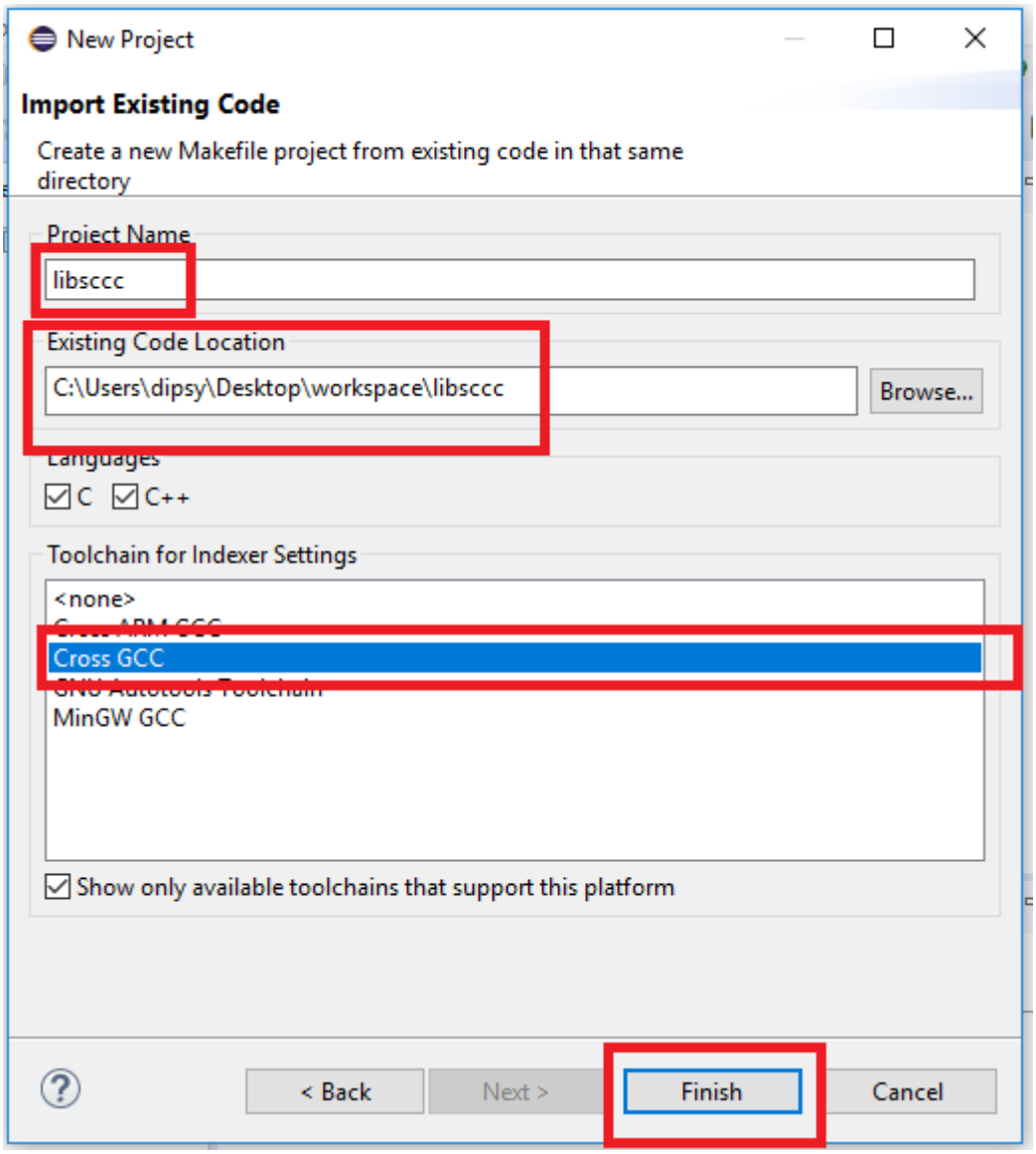


- RightClick blank space on [Project Explorer] tab

- [C/C++ : Existing Code as Makefile Project] -> [Next]

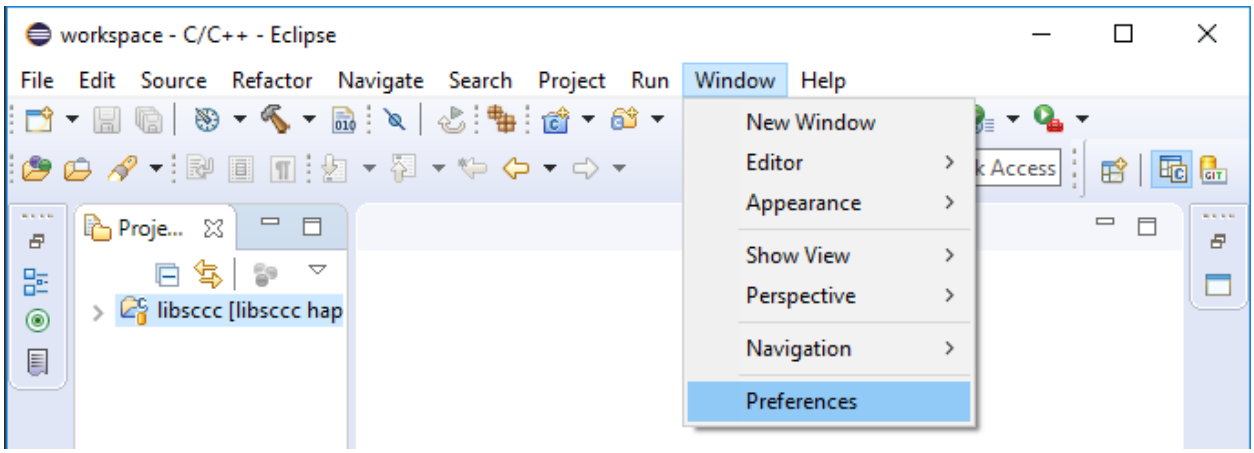


- Select Cross GCC
- Name the new project as `libsccc`
- Set the existing code location
- [Finish]

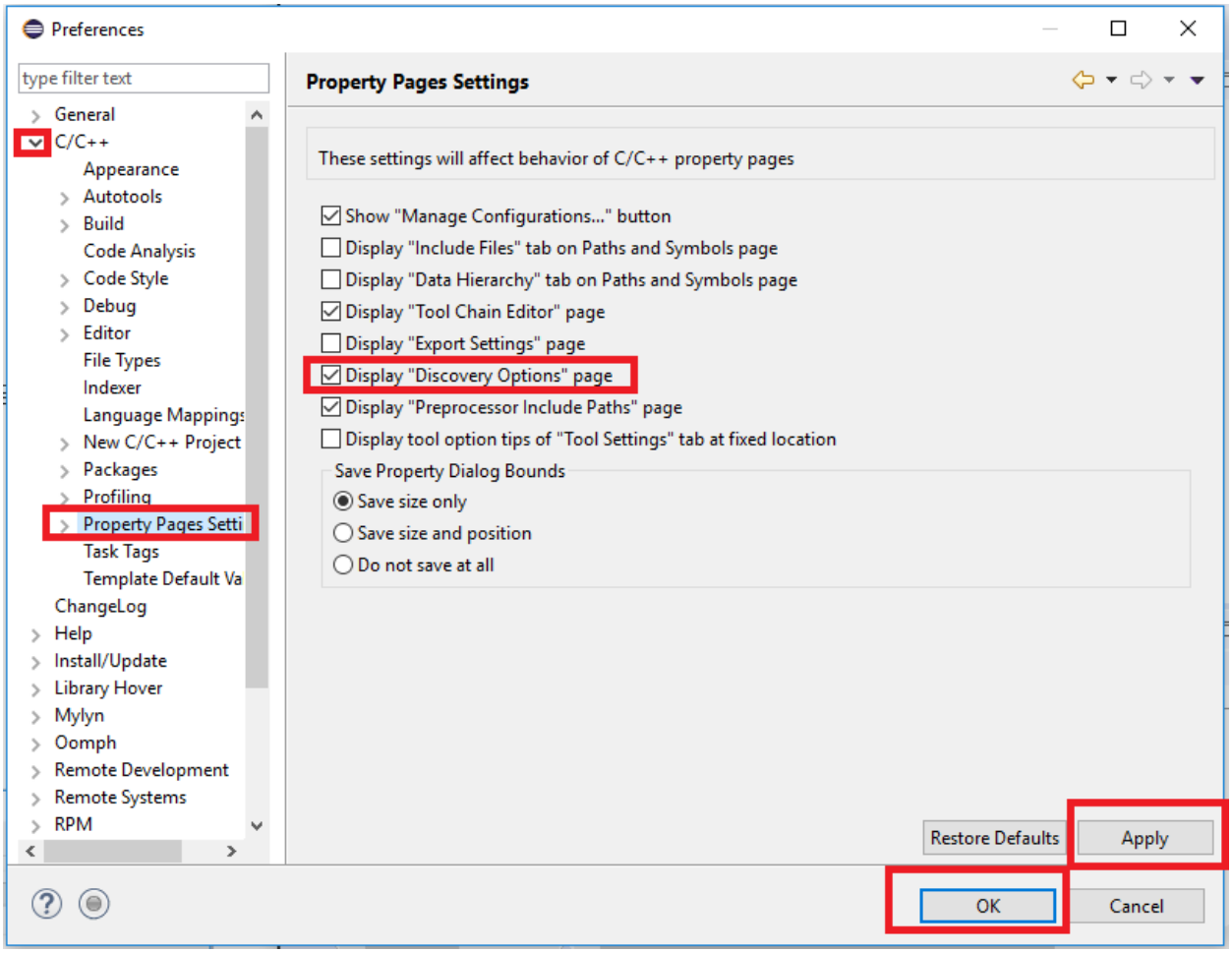


3.3 Discovery Option Setting

- [Window] -> [Preferences] -> [C/C++ : Property Pages Settings]



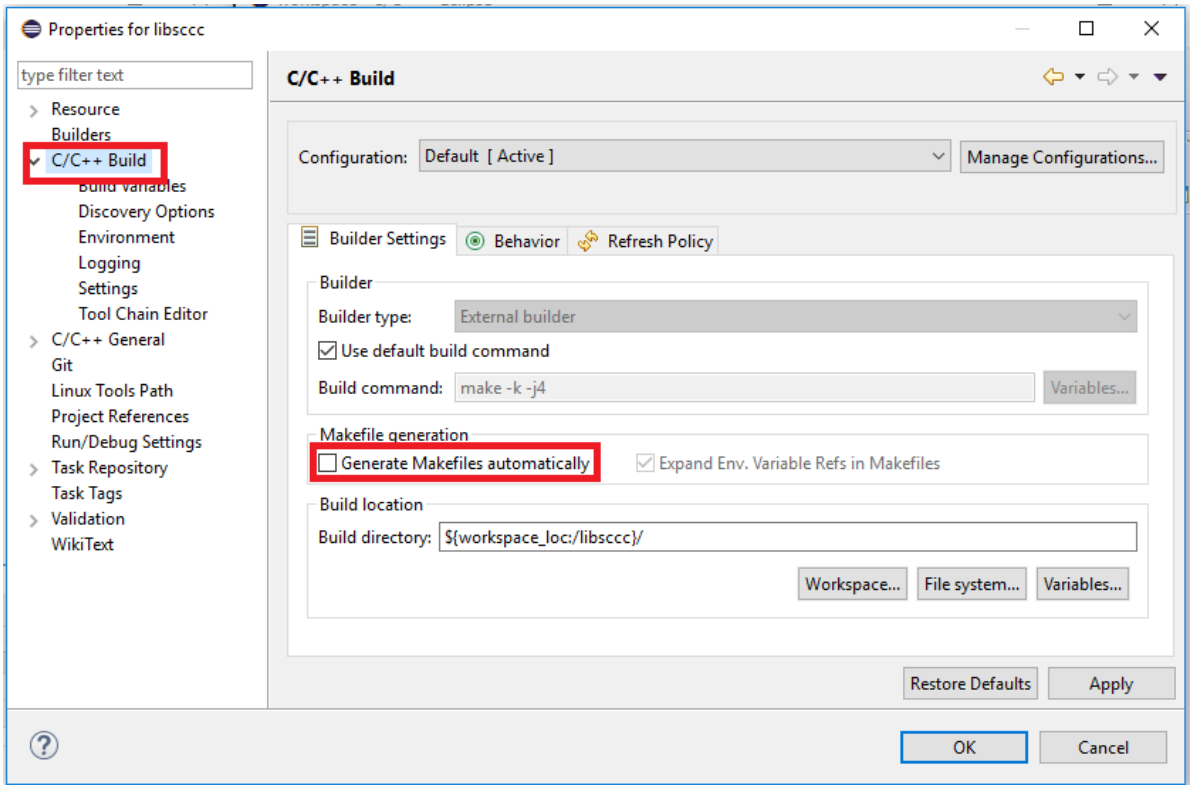
- Check [Display “Discovery Options” page]
- [Apply]
- [OK]



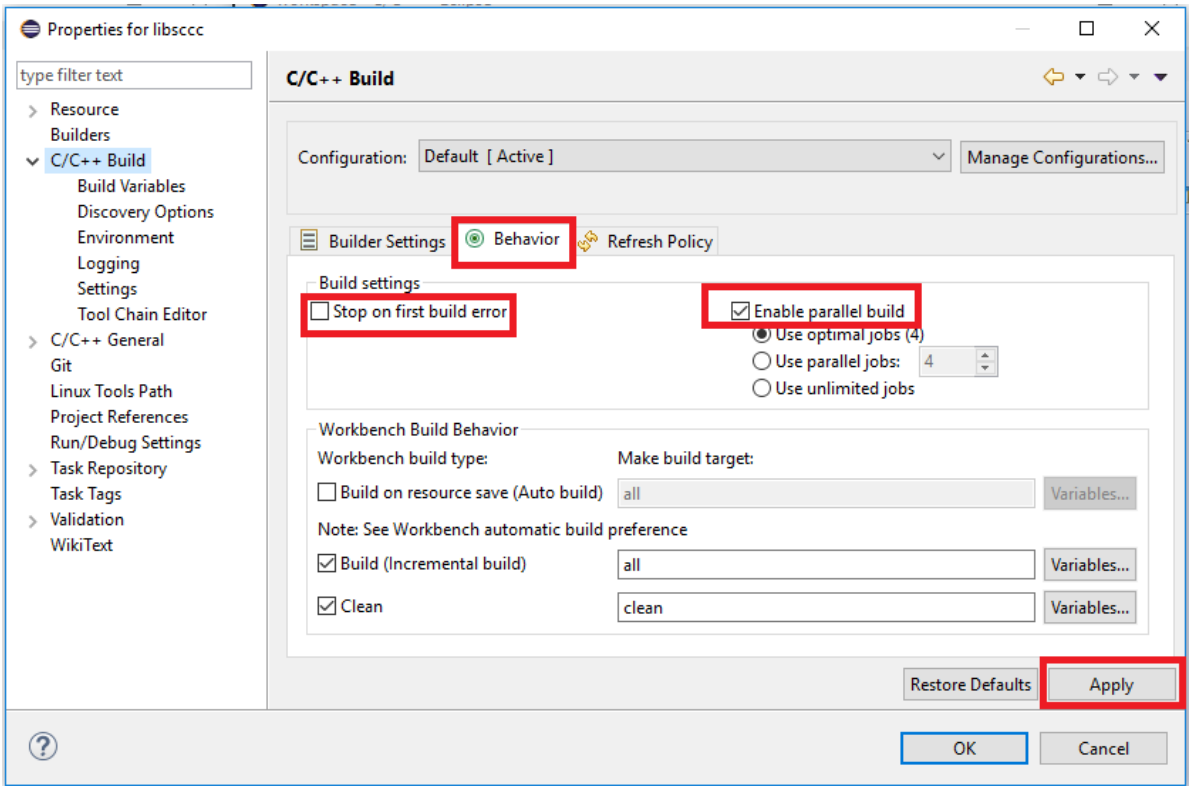
3.4 Properties Setting

https://youtu.be/v834HF_Uha4?t=5m

- RightClick [libsccc] on [Project Explorer] tab -> [Properties]
- Goto [C\C++ Build]
    - Uncheck [Generate Makefiles automatically]

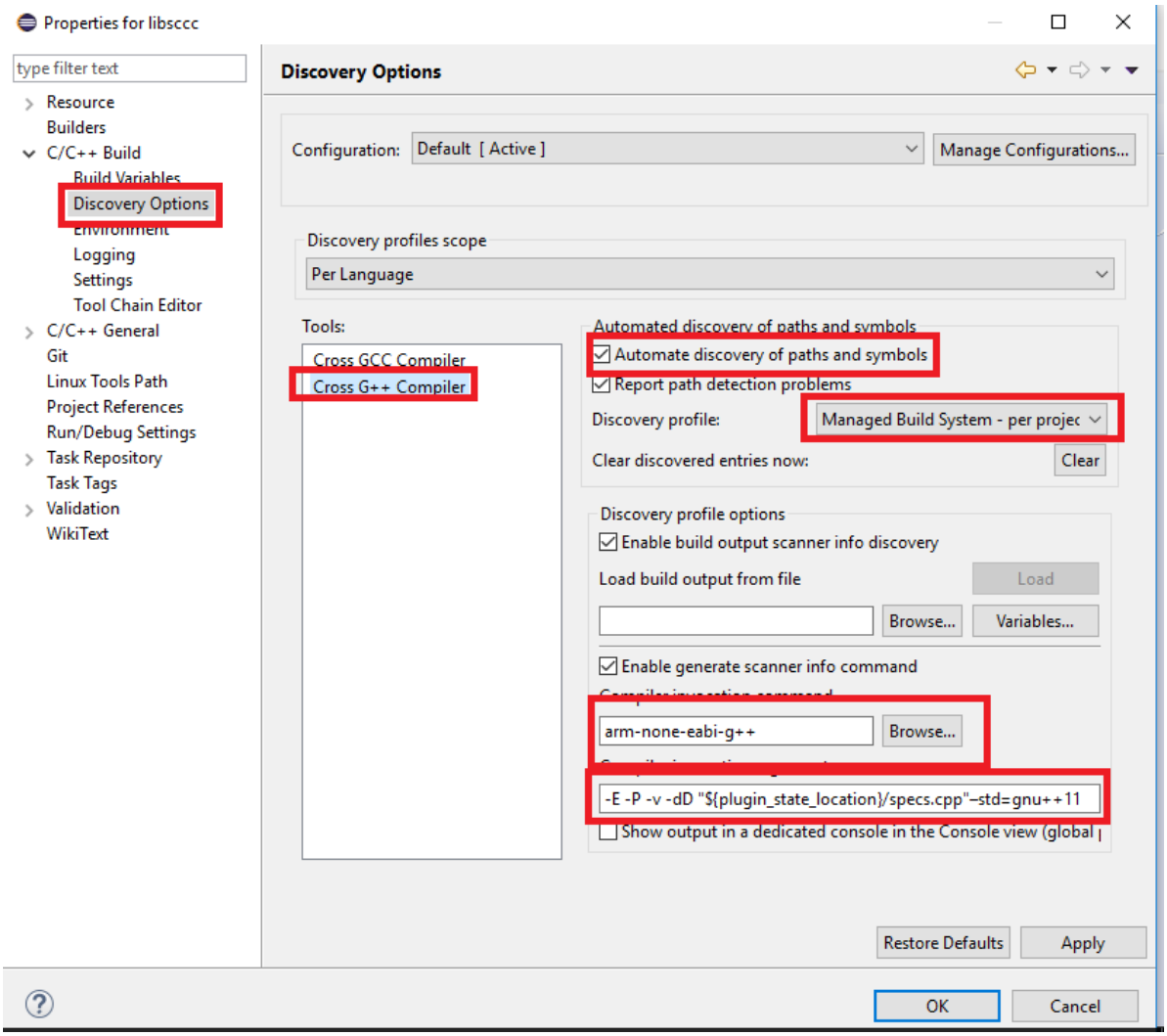


- Goto [Behavior] tab
  - Uncheck [Stop on first build error]
  - Check [Enable parallel build] (Optional)
- [Apply]



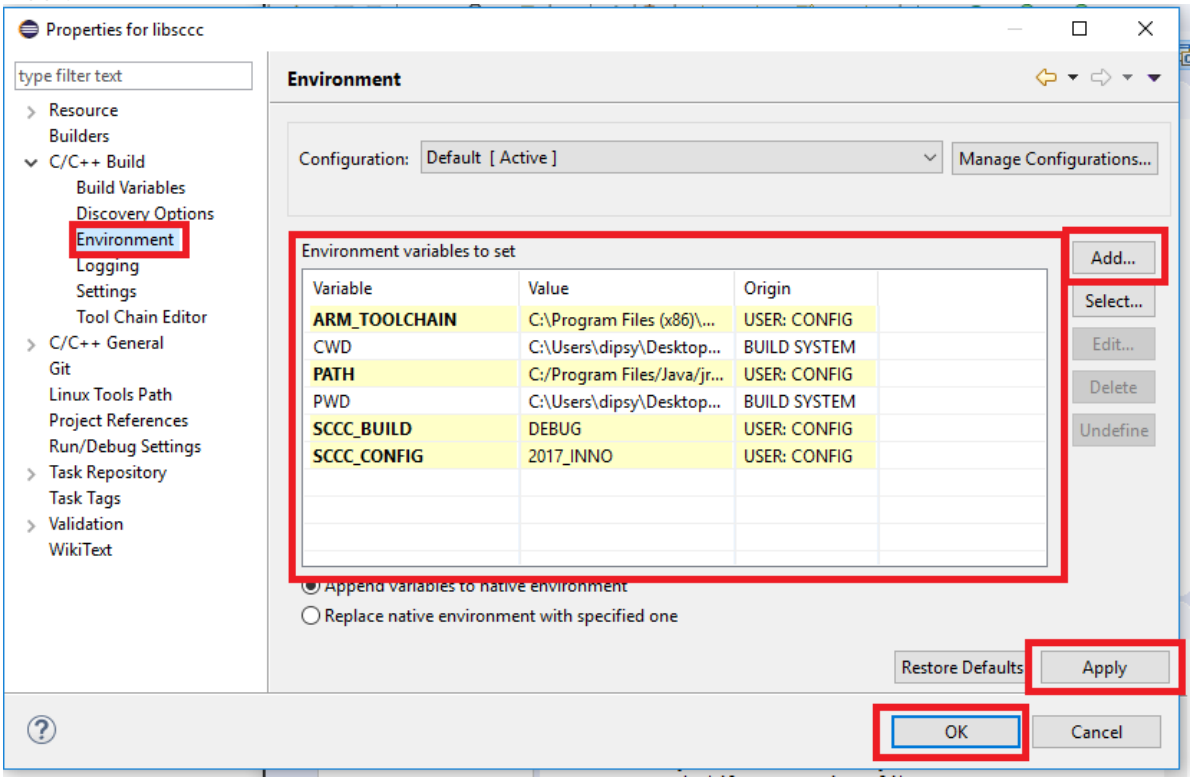
- Goto [C/C++ Build : Discovery Options]
  - Tools: [Cross G++ Compiler]
  - Check [Automate discovery of paths and symbols]

        - Ignore the warning "this discovery method is deprecated" if any
          (This warning will appear if you are using Oxygen)
    - Discovery profile: [Managed Build System - ...]
    - Compiler invocation command : `arm-none-eabi-g++`
    - Append `-std=gnu++11` to Compiler invocation argument
    - If you can's see the options, make the window full screen

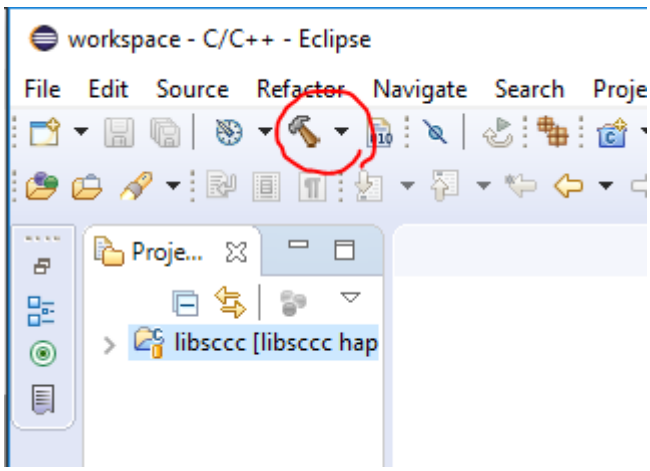


- Goto [C/C++ Build : Environment]
    - [Add...] -> Name: `ARM_TOOLCHAIN` -> Value: `{ Path of GNU Tools for ARM folder }`
    - [Add...] -> Name: `SCCC_BUILD` -> Value: `DEBUG`
    - [Add...] -> Name: `SCCC_CONFIG` -> Value: `2017_INNO`
    - Append `{ Path of GNU Tools for ARM folder }\bin` to `[PATH]` Variable
    - Append `{ Path of GNU Make folder }\bin` to `[PATH]` Variable

- [Apply]



- **Build the library**

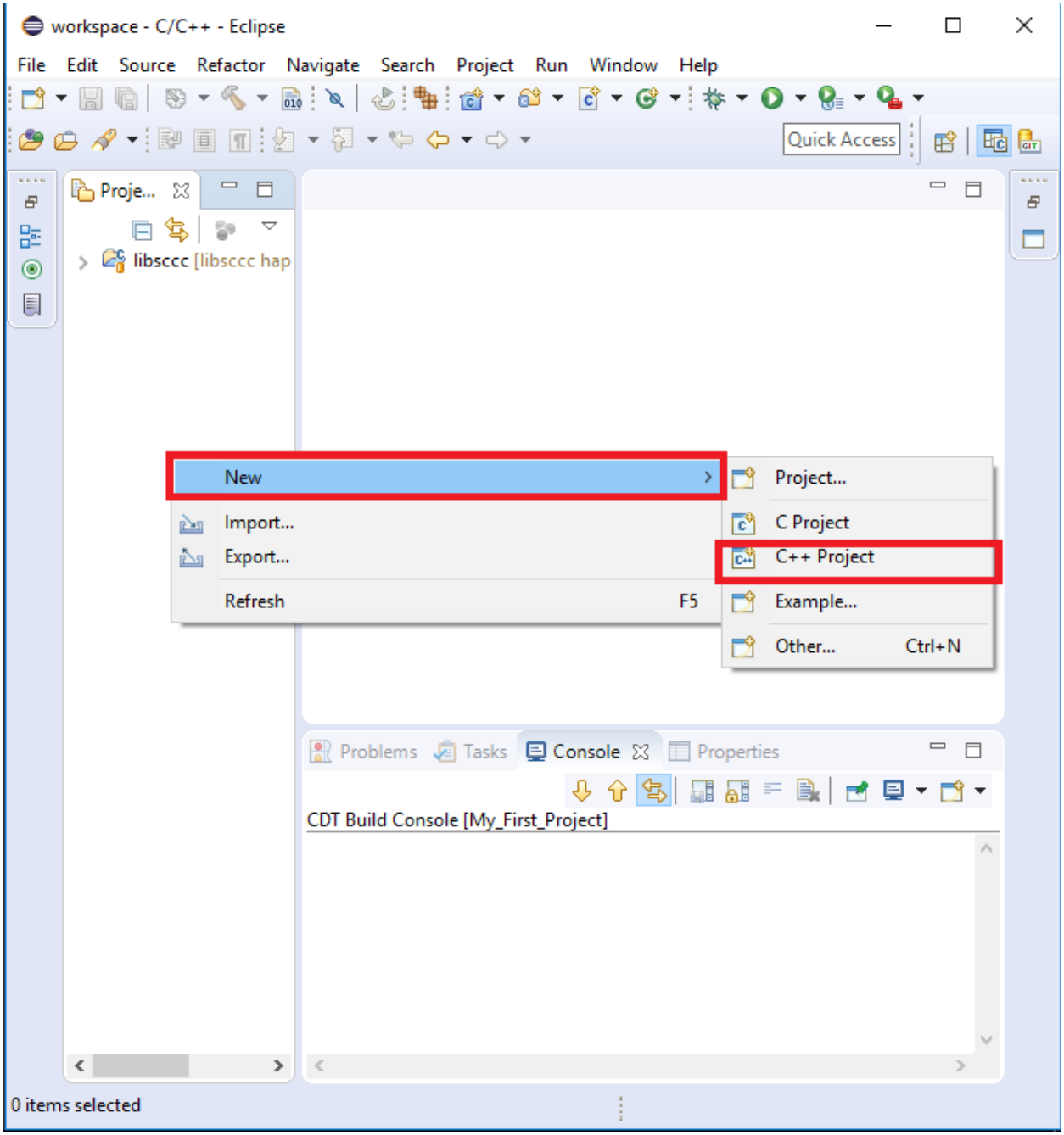


4. Project Setting

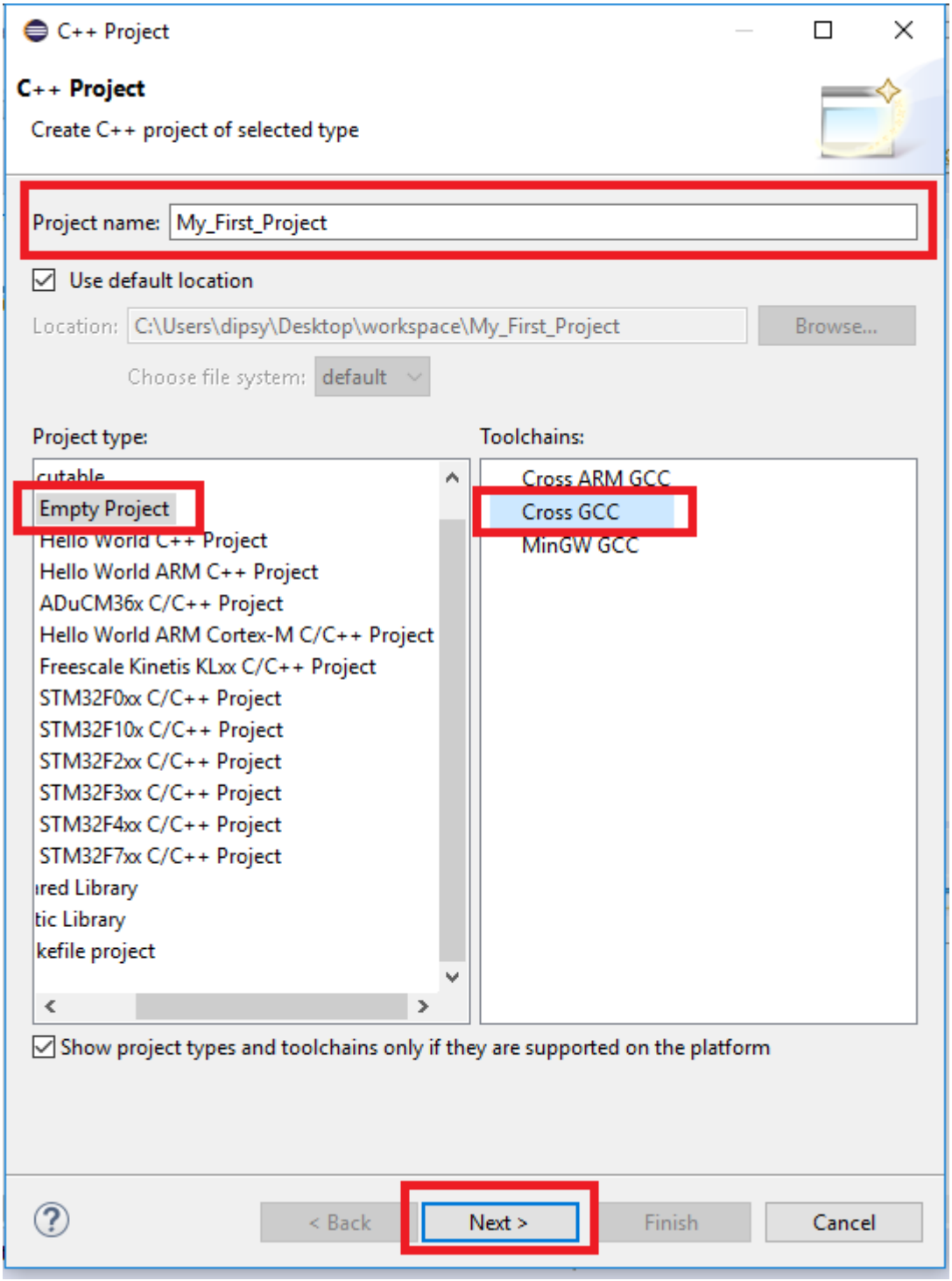
whenever you want to create a new project for programming smart car, you can start from this section. It is always a good measure to have a spare project for testing. https://youtu.be/v834HF_Uha4?t=9m

4.1 Create New Project

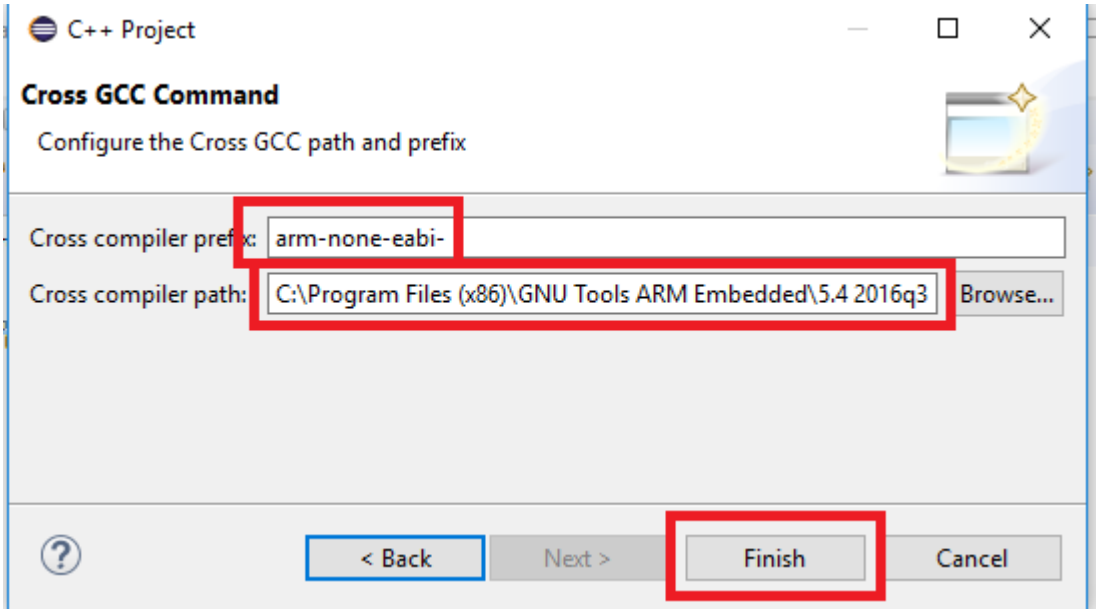
- Right-click on [Project Explorer] -> [New] -> [C++ Project]



- Choose project name, don't include white space or you will GG
- Project Type: [Executable : Empty Project], Toolchains: [Cross GCC]



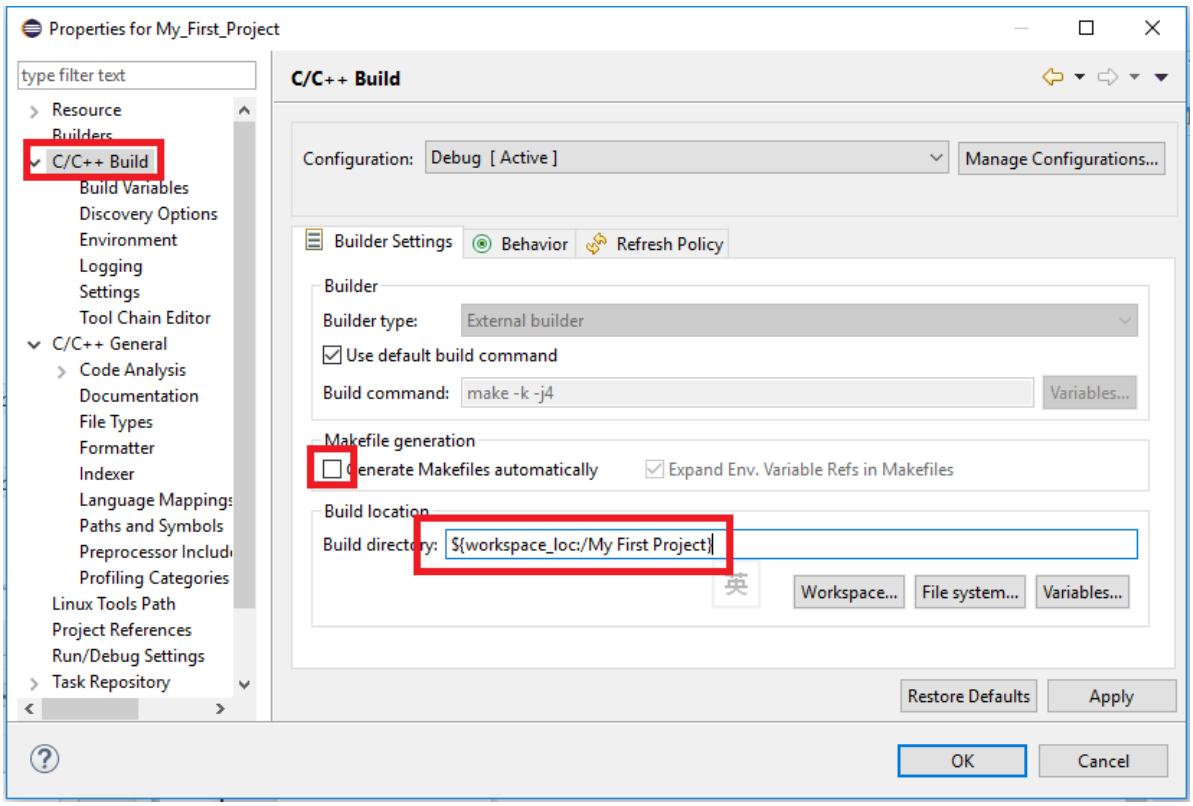
  - -> [Next] -> [Next]
  - Cross compiler prefix: `arm-none-eabi-`
  - Cross compiler path: `{ Path to GNU Tools for ARM }`
- [Finish]



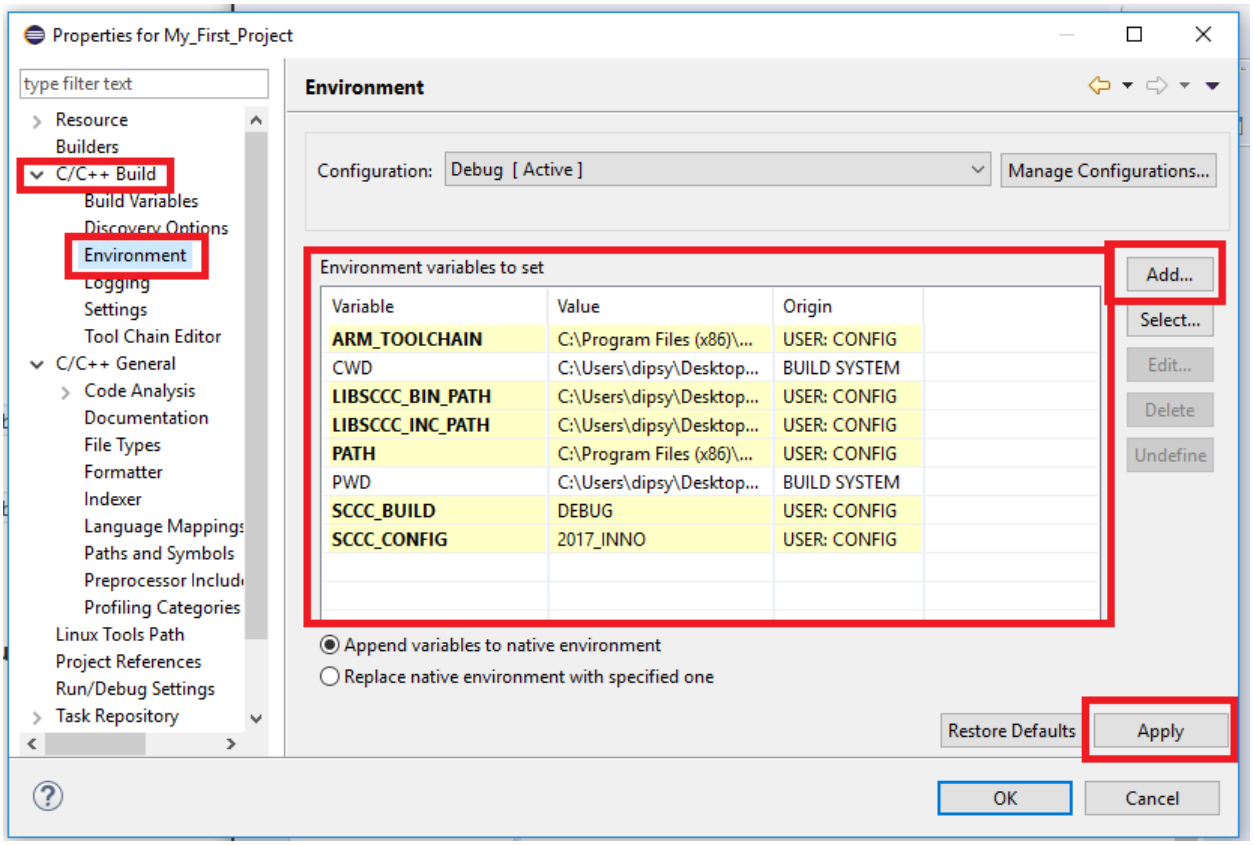
4.2 Property Setting

Goto [C/C++ Build]

- Goto [Builder Settings] tab
  - Remove `Debug` from Build directory



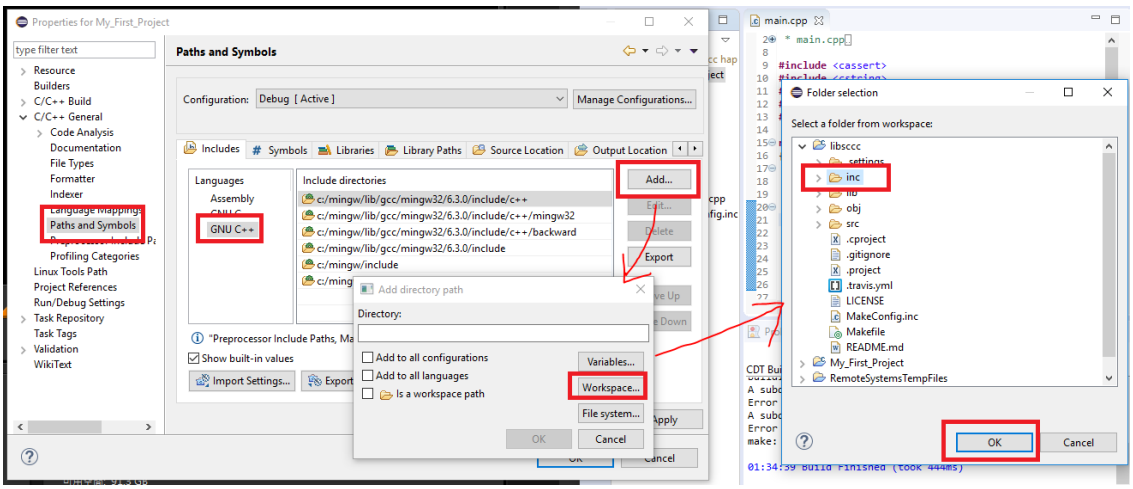
- repeat 3.4
- Goto [C/C++ Build : Environment]
  - [Add...] -> Name: `LIBSCCC_BIN_PATH` -> Value: `{ Path of library folder }\lib`
  - [Add...] -> Name: `LIBSCCC_INC_PATH` -> Value: `{ Path of library folder }\inc`
- [Apply]



Remember that variable [SCCC_CONFIG] of both library and your own project must be the same.

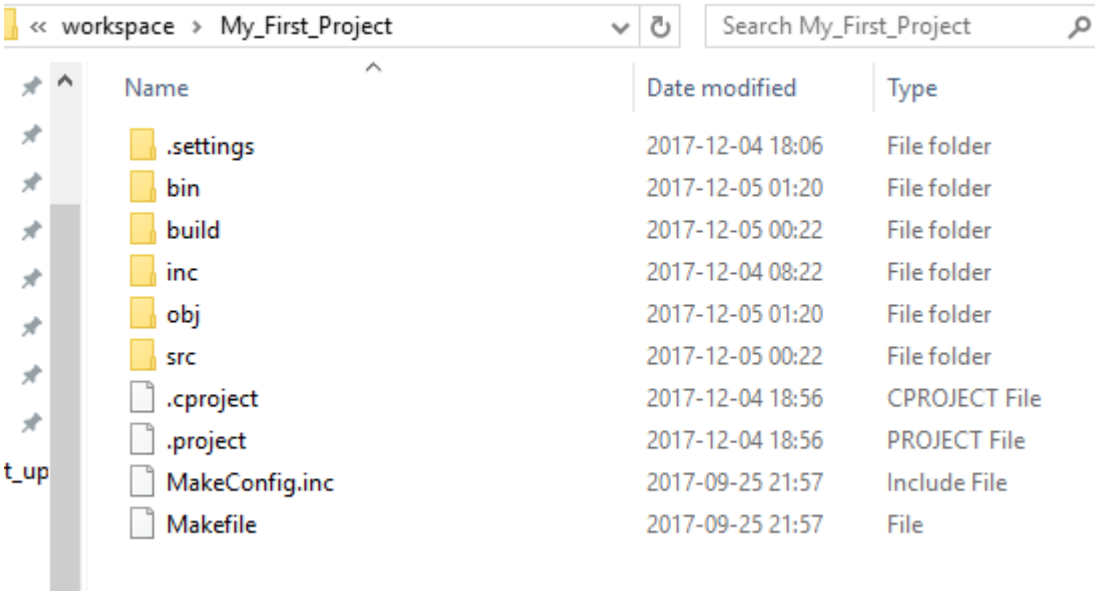
4.3 Include the Library to your Project

- Goto [C/C++ General : Paths and Sybmols]
  - Goto [Includes] tab
    - Select Languages: [GNU C++]
    - [Add....] -> [Workspace...] -> Select [libsccc : inc]-> [OK] -> [OK]



4.4 Download linker's script, makefile and main.cpp

- https://drive.google.com/drive/u/0/folders/0B-KAbvJ3lRuIREhIQzllSy1iZjQ
- Download, extract and put to your root folder



- Don't change this part unless you know what you are doing

```
namespace libbase {
namespace k60 {
Mcg::Config Mcg::GetMcgConfig() {
  Mcg::Config config;
  config.external_oscillator_khz = 50000;
  config.core_clock_khz = 150000;
  return config;
}
}  // namespace k60
}  // namespace libbase
```

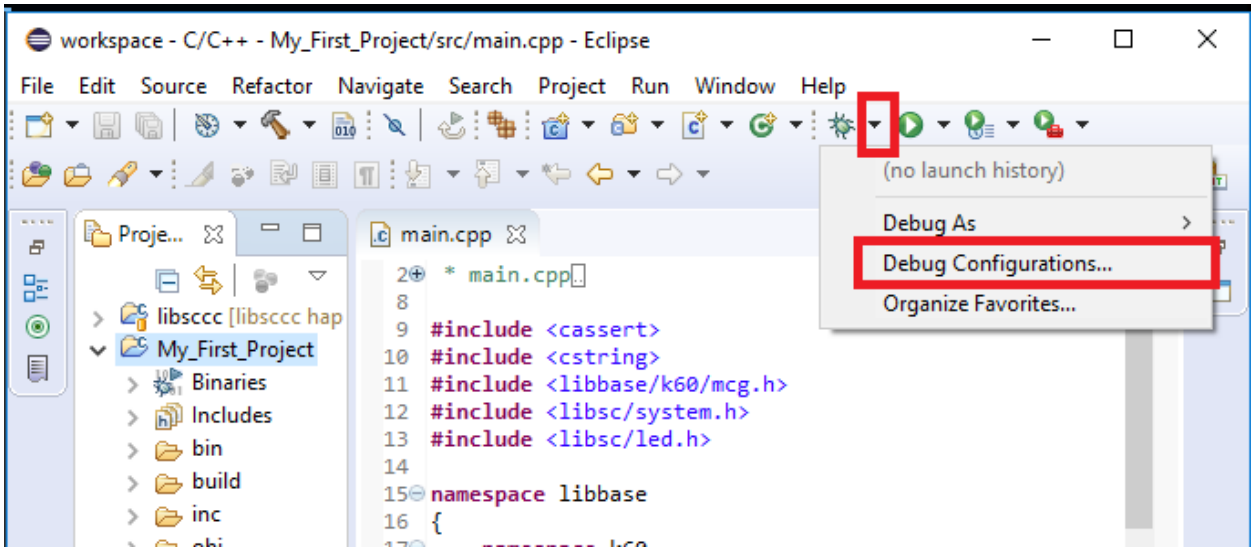
- **Building the codes**

5 Debugger setting

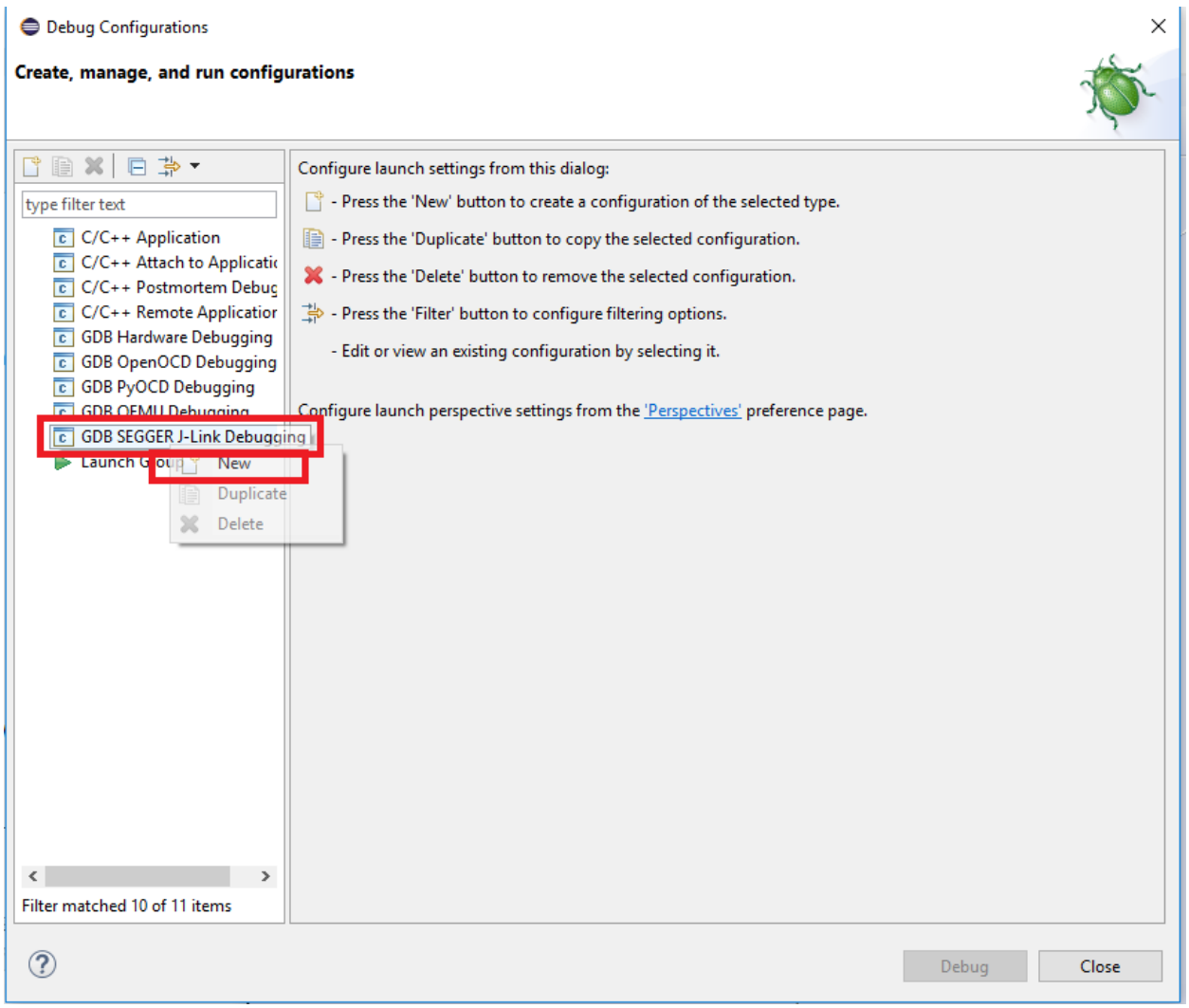
(After you finish **building the project** without error and **refresh project**) https://youtu.be/v834HF_Uha4?t=17m27s

5.1 New Debug Config

- Press the triangle next to a “BUG”
- [Debug Configurations...]

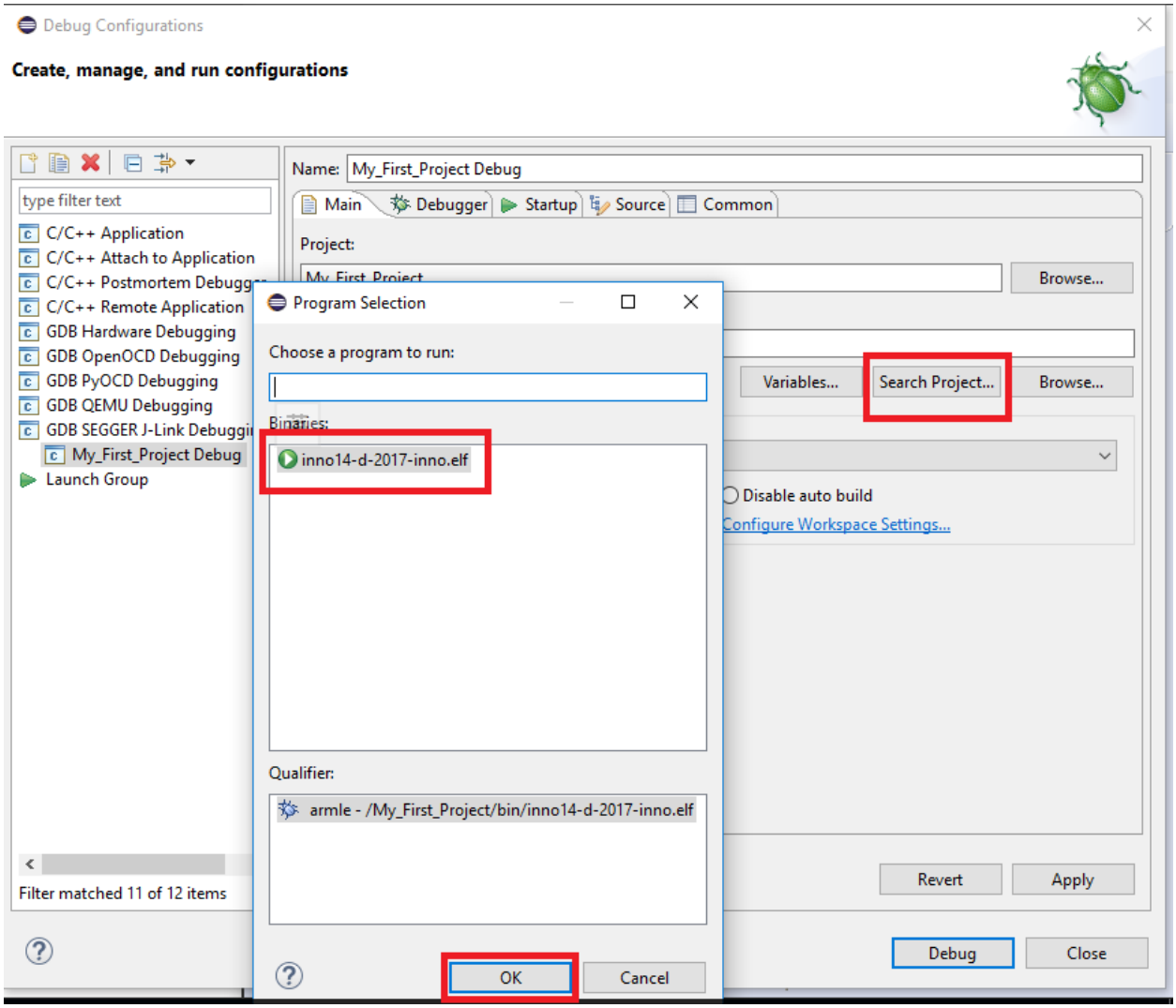


- RightClick [GDB SEGGERJ-Link Debugging
- [New]

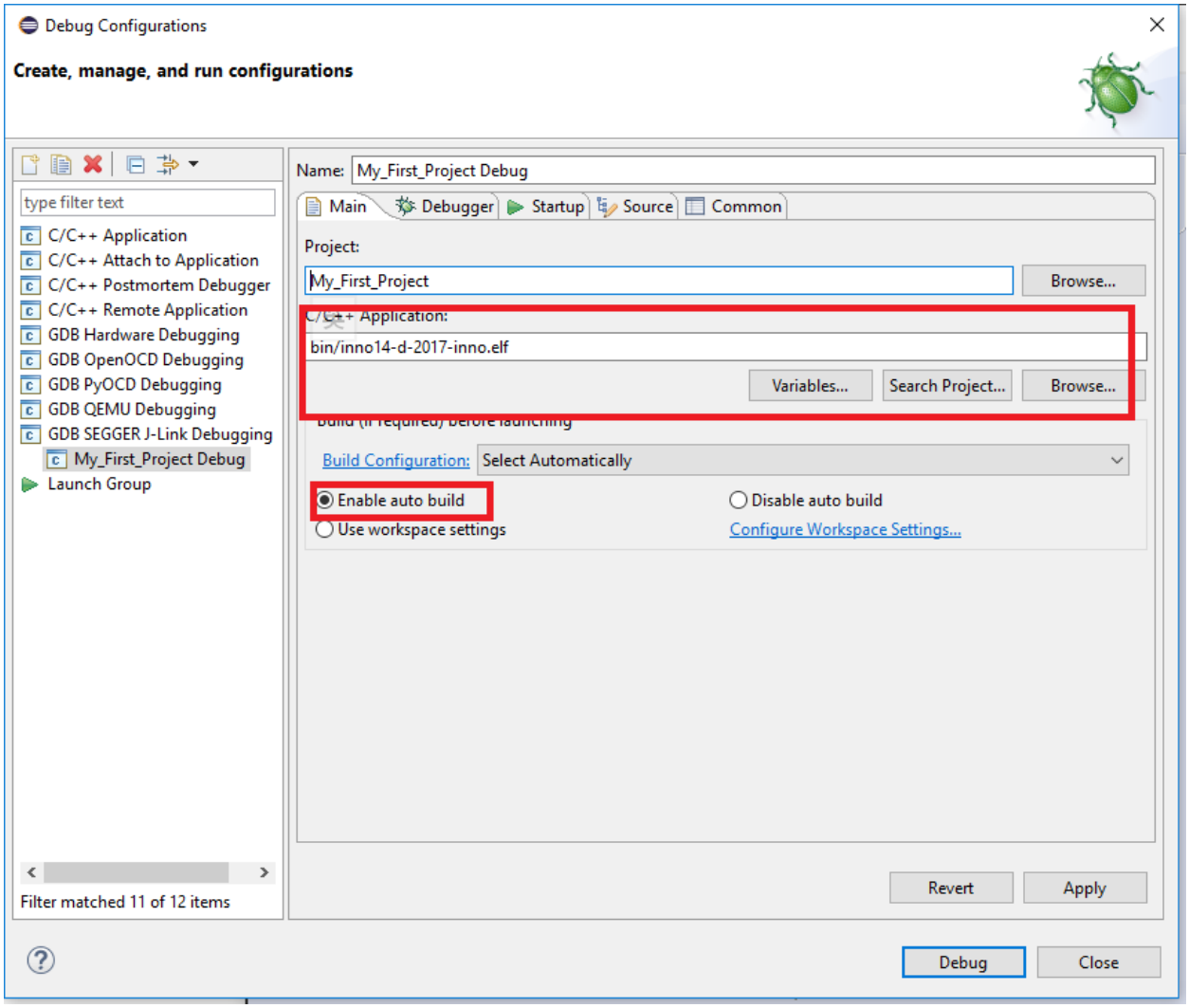


5.2 Select Binary

- You should find something like `bin/??????.elf` here (This will be automatically generated upon successful project build)
- If not, you should press [Search Project...] and find it

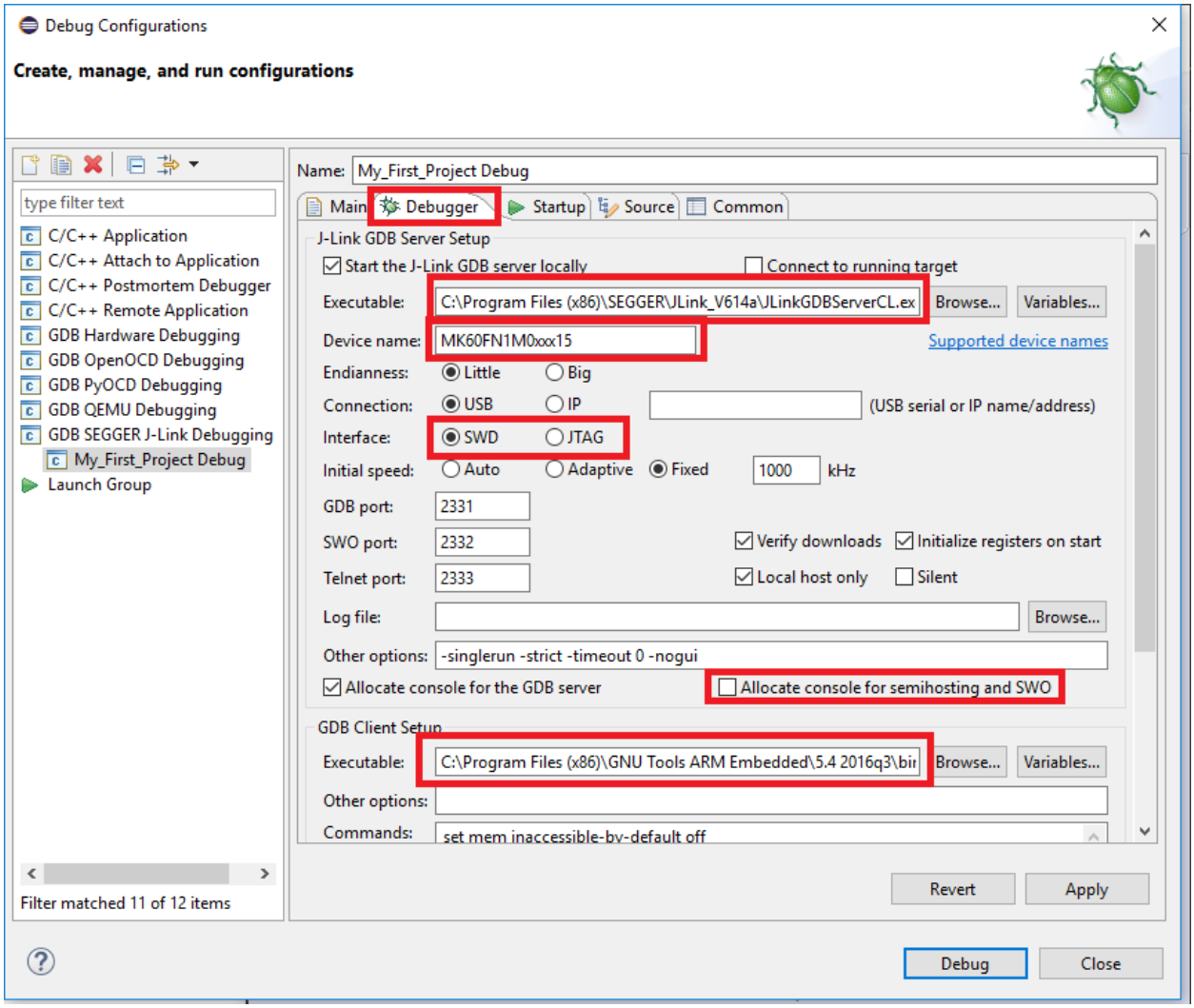


- If you still cannot find it, **refresh the project** by clicking F5, and see whether there is a binary directory at the project explorer
- Then, you should enable auto build because there is always someone who debugs a million times without rebuilding the project. XD



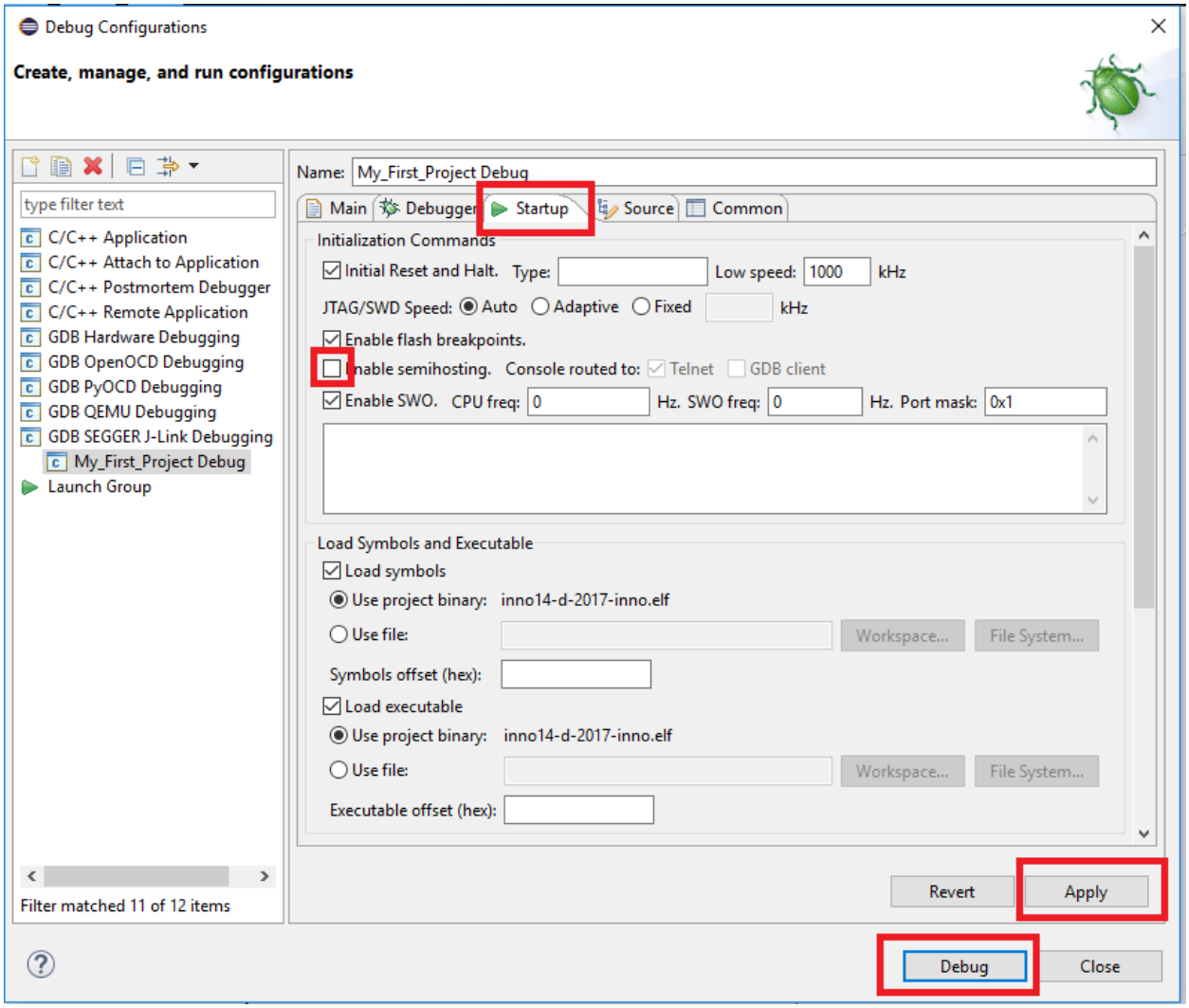
5.3 Select MCU and J-Link

- Goto [Debugger] tab
- Server Executable: `{path to JLink}\JLinkGDBServerCL.exe` (For window, it is under C:\Program Files (x86)\SEGGER\<Jlink version>\JLinkGDBServerCL.exe)
- Device name: `MK60FN1M0xxx15` (MCU)
- SWD (small J) or JTAG(J-Link) (big J)
- Uncheck [Allocate console for semihosting and SWO]
- Client Executable: `{path to GNU for ARM}\bin\arm-none-eabi-gdb.exe`



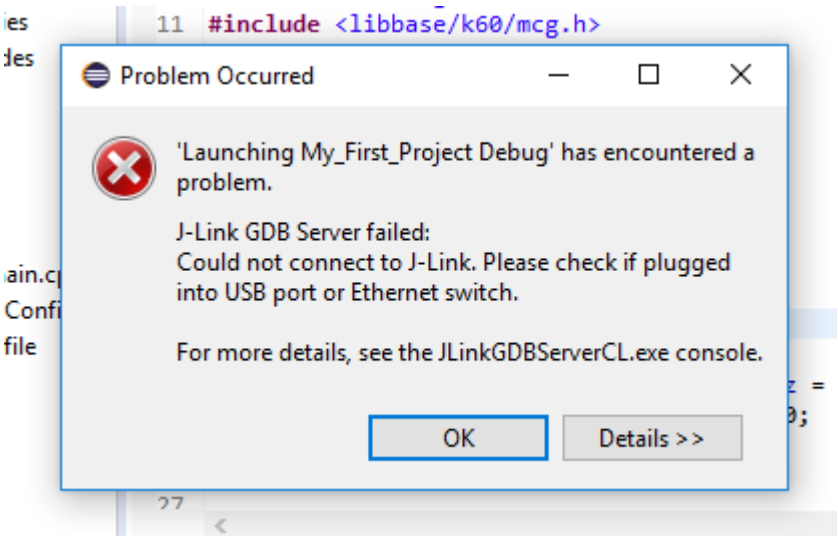
5.4 No Semi-hosting

- Goto [Startup] tab
- Uncheck [Enable semihosting]
- [Apply]
- [Debug]



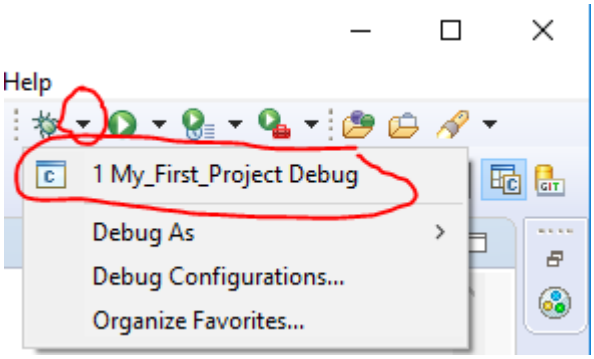
5.5 Wait for Flashing

If you see this error message, it means you successfully set up eclipse (this error means everything is successfully built, but JLink server cannot find a connected JLink)



6. Done and Enjoy Your SmartCar Journey 😘

Whenever you want to flash program, clean project -> build project -> click the debugger



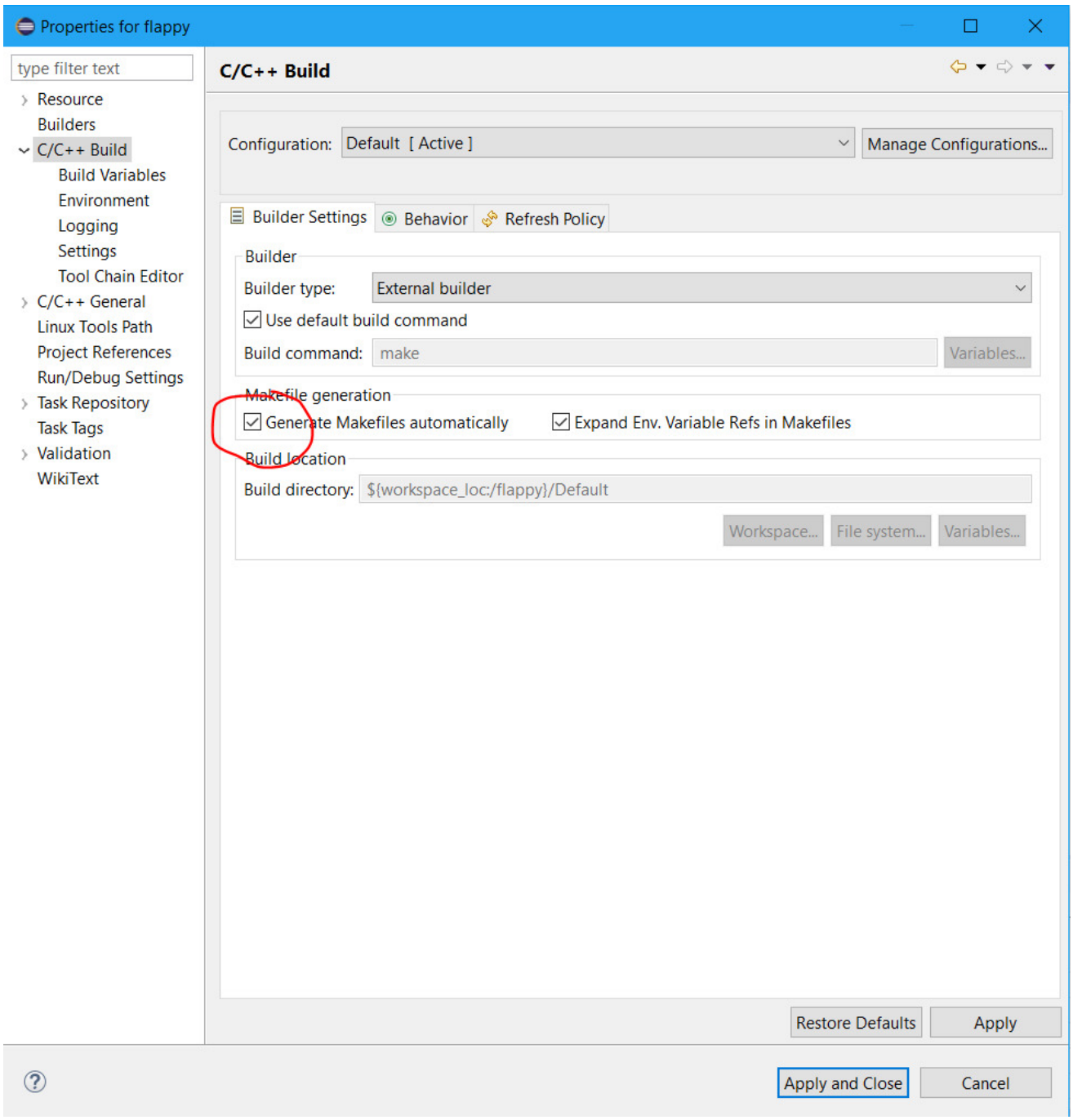
Troubleshooting

1. G++ GCC not found (By Daniel Cheung):

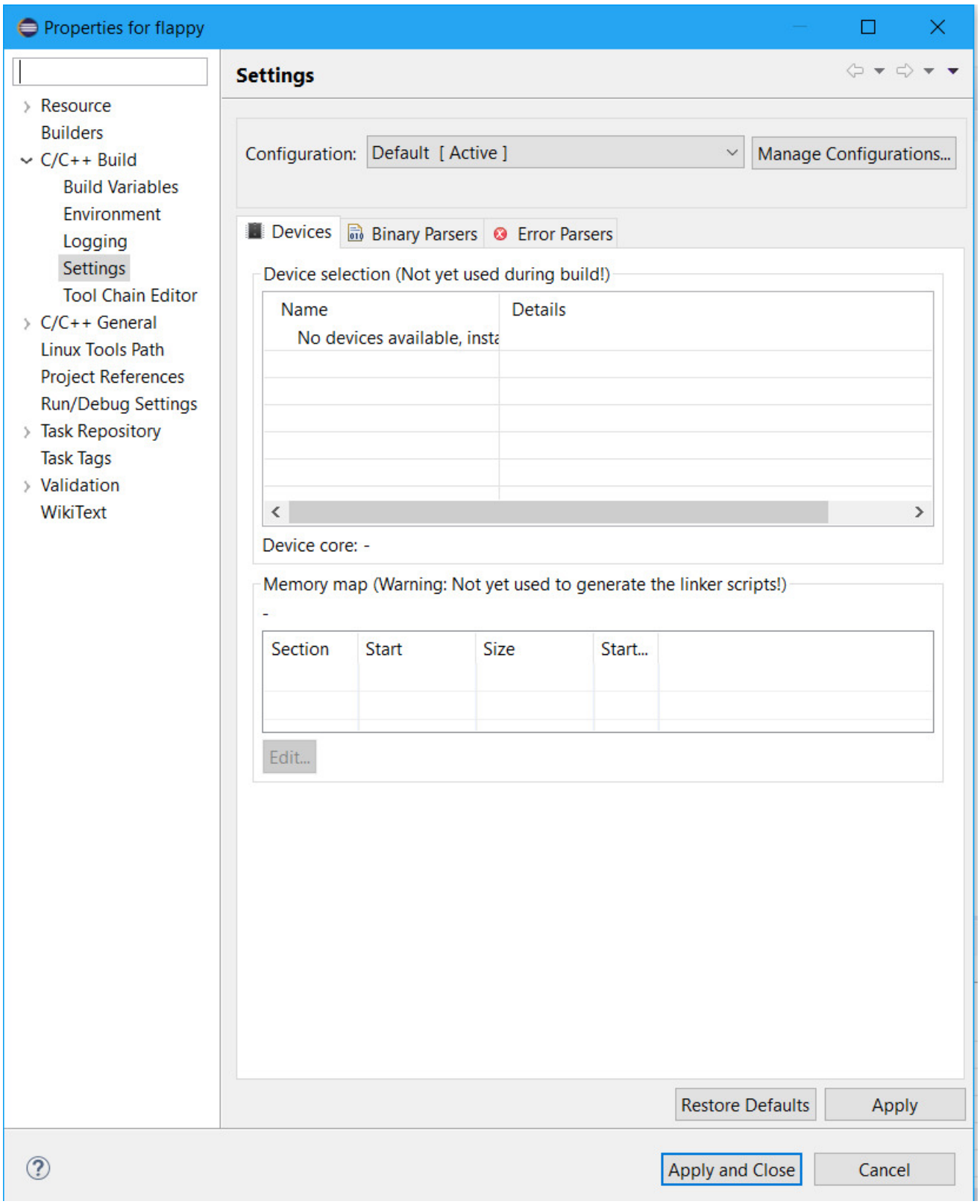
Eclipse is just going to try using the normal g++

The way to fix it, is to do the following steps

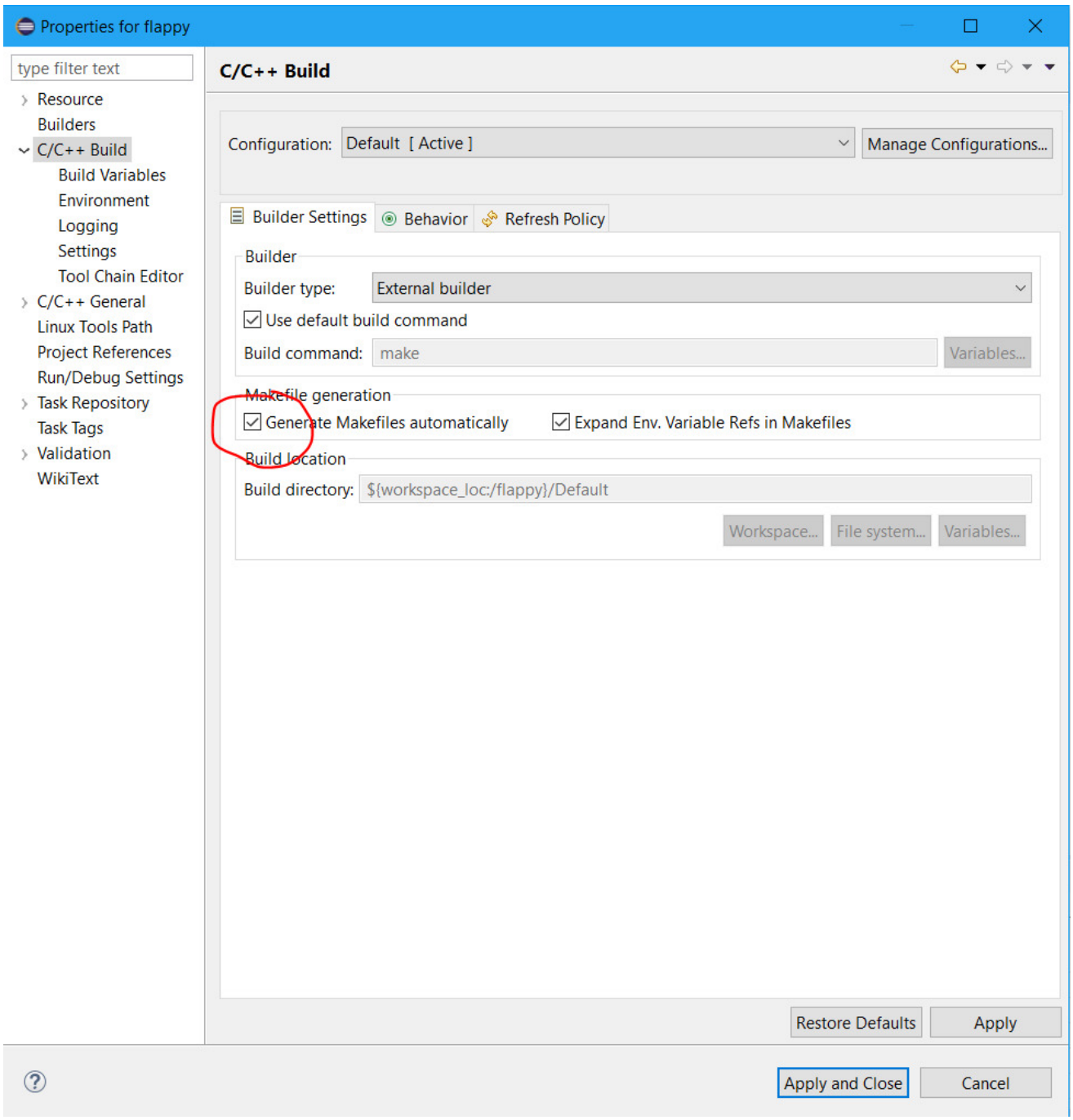
The problem is you will not be able to change the prefix with some settings because the tools settings will disappear, you need to check to automate make first, then change the tool settings and confirm and change back the auto make to manual



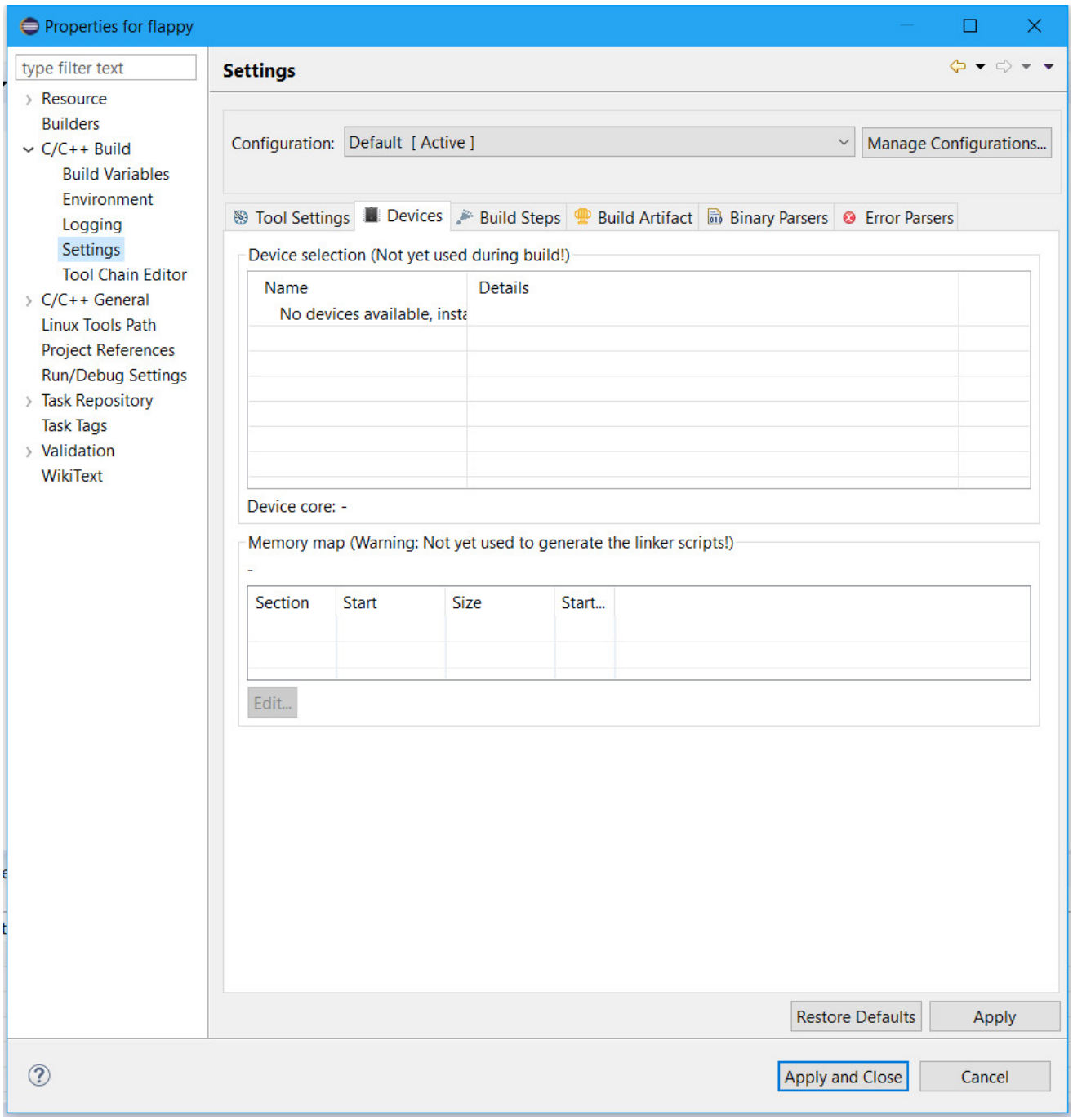
More explanations: ok by default, this screen looks like this:



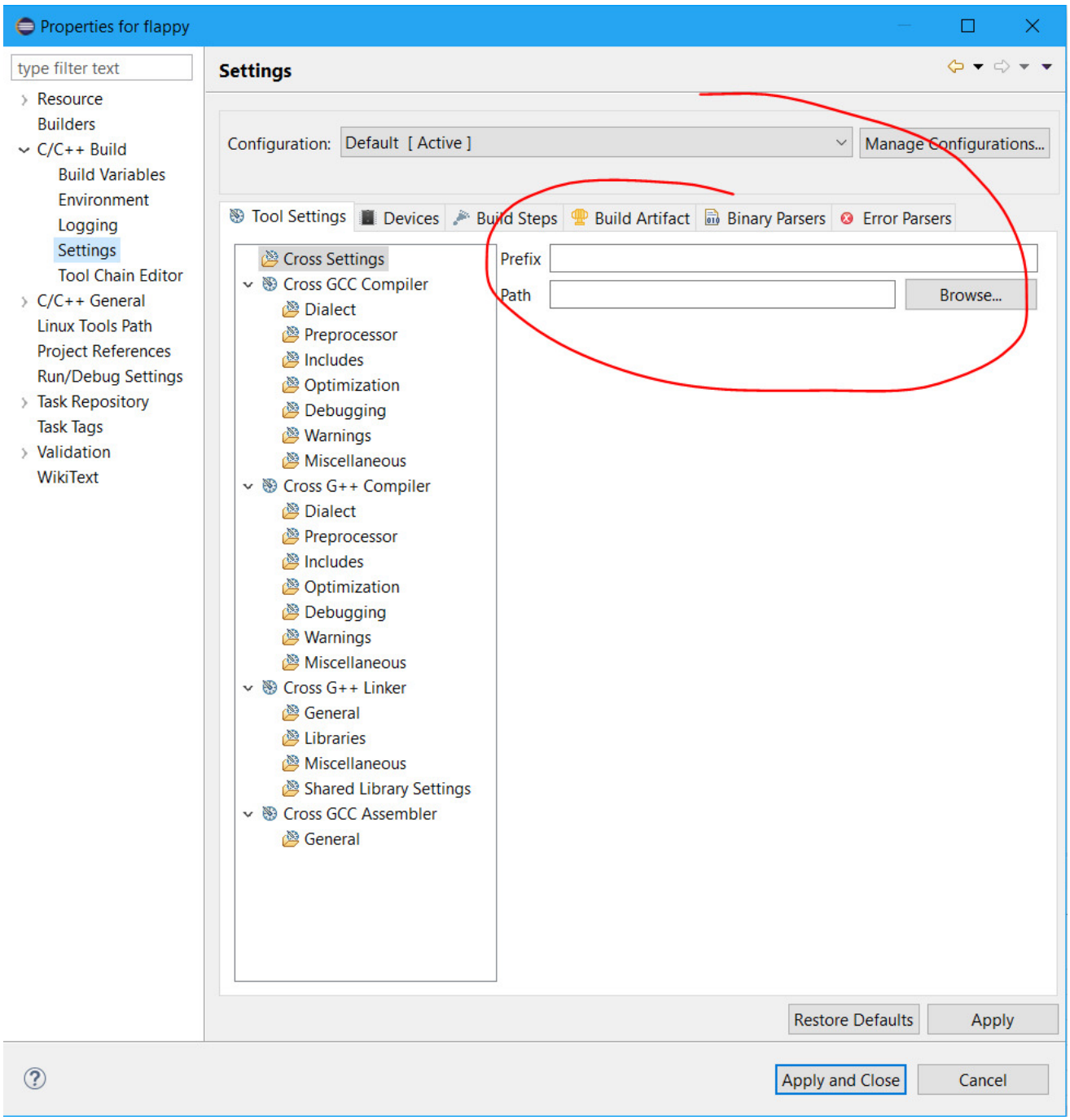
But, if you uncheck this:



the page reveals more option



this being empty is probably the culprit



So, this implies that the MakeFile was supposed to be able to indicate the prefix for the compiler, but it didn't for some reason.

I think that's why Eclipse decided to hide these tabs, because they were supposed to be decided in the MakeFile (me leering at whoever made the MakeFile)

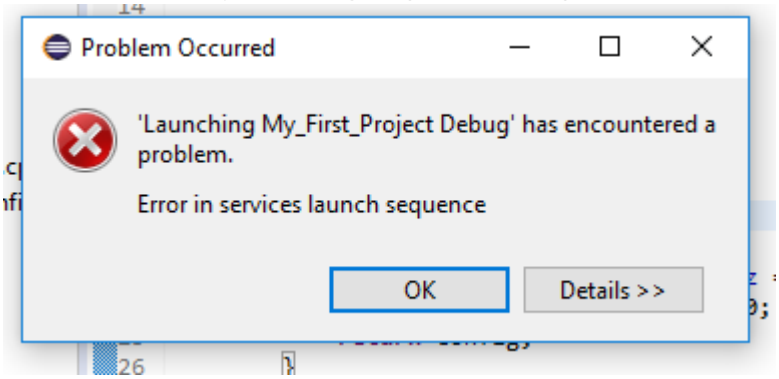
2. Whenever you fail flashing, try this:

1. Refresh project
2. Clean project
3. Build project
4. Debug

If still fail, try building the library again, and repeat steps above

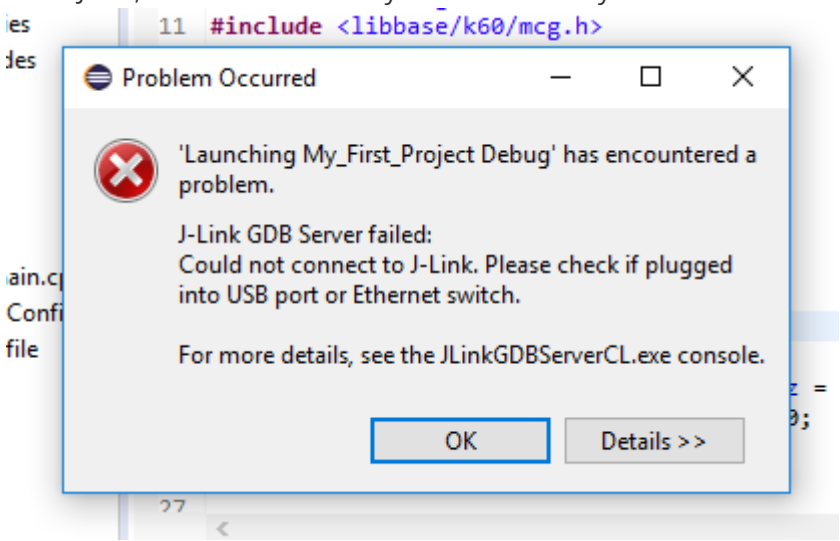
3. If launching sequence error,

Unplug and plug the USB if you are flashing program. If still cannot solve, it may be you can config the debugger wrongly. If still cannot solve, try restarting eclipse and computer.



4. If GDB server fail,

check your connection of JLink, and check whether you have turn on your smart car



Care only errors, ignore the warnings