

# Remote Debugging with Eclipse

MA35D1 | NUC9XX

江天文

2024/01/25

Joy of innovation  
**nuvoTon**

# | Introduction

- The Eclipse Public License is designed to be a business-friendly free software license, and features weaker copyleft provisions than licenses such as the GNU General Public License (GPL).

# | Installing Eclipse

- Download **Eclipse Installer 2013-12 R** from the site <https://www.eclipse.org/downloads/packages>
- Extract the package **eclipse-inst-jre-linux64.tar.gz**  
`$ tar -xf eclipse-inst-jre-linux64.tar.gz 2> /dev/null`
- Execute the installer **eclipse-installer**  
`$ cd eclipse-installer`  
`$ ./eclipse-inst`

The Eclipse Installer 2023-12 R now includes a JRE for macOS, Windows and Linux.

## Try the Eclipse **Installer** 2023-12 R

The easiest way to install and update your Eclipse Development Environment.

[Find out more](#)

📄 874,864 Installer Downloads

📄 772,712 Package Downloads and Updates

### Download

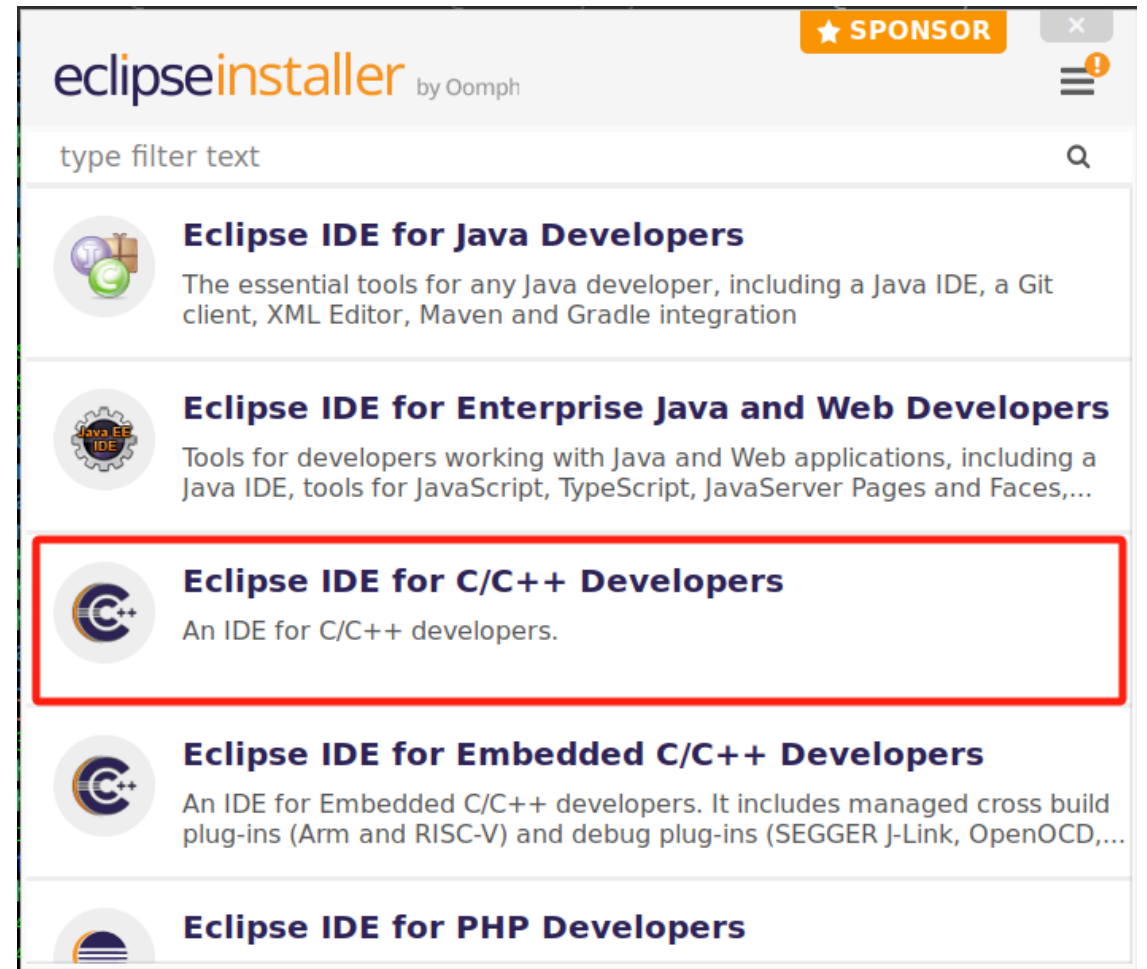
macOS [x86\\_64](#) | [AArch64](#)

Windows [x86\\_64](#)

Linux [x86\\_64](#) | [AArch64](#)

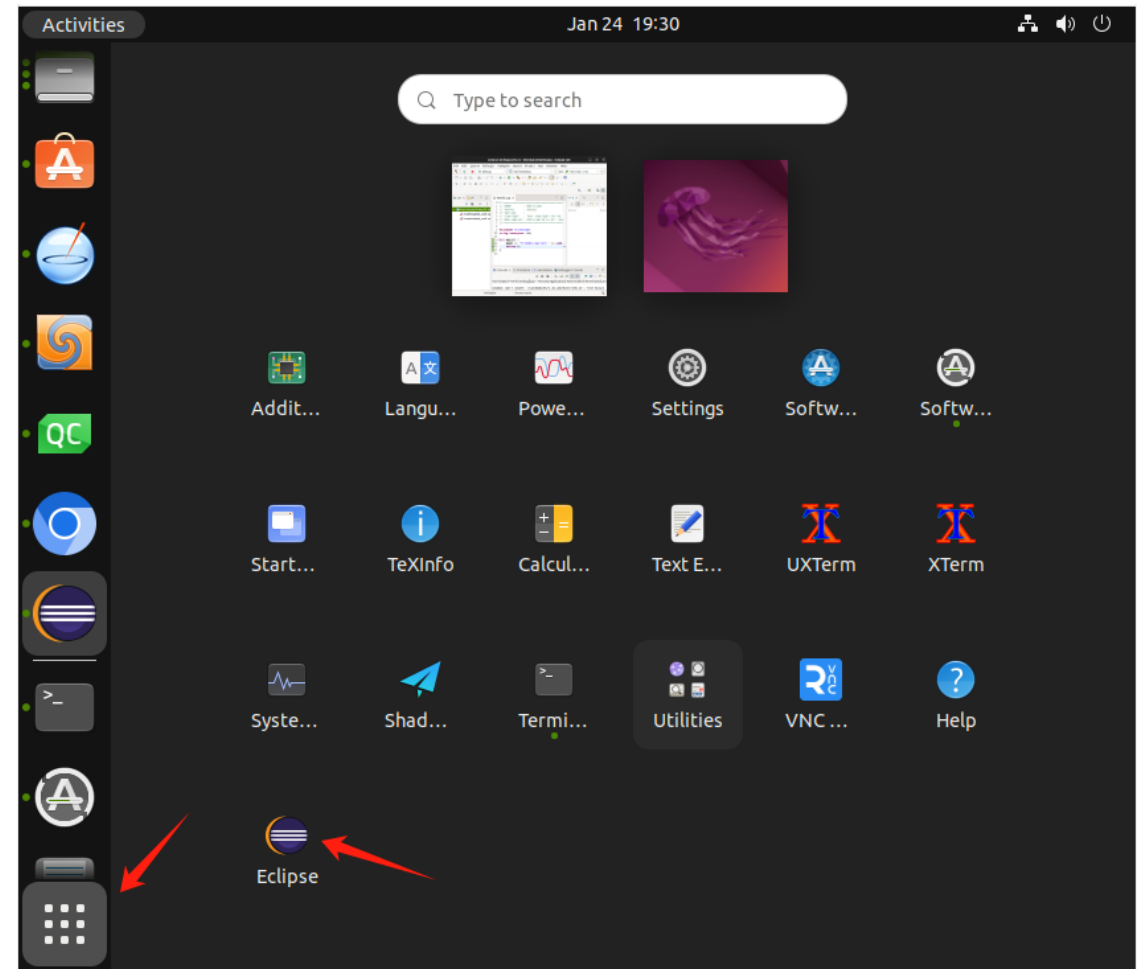
# | Installing Eclipse

- Choose the *Eclipse IDE for C/C++ Developers*



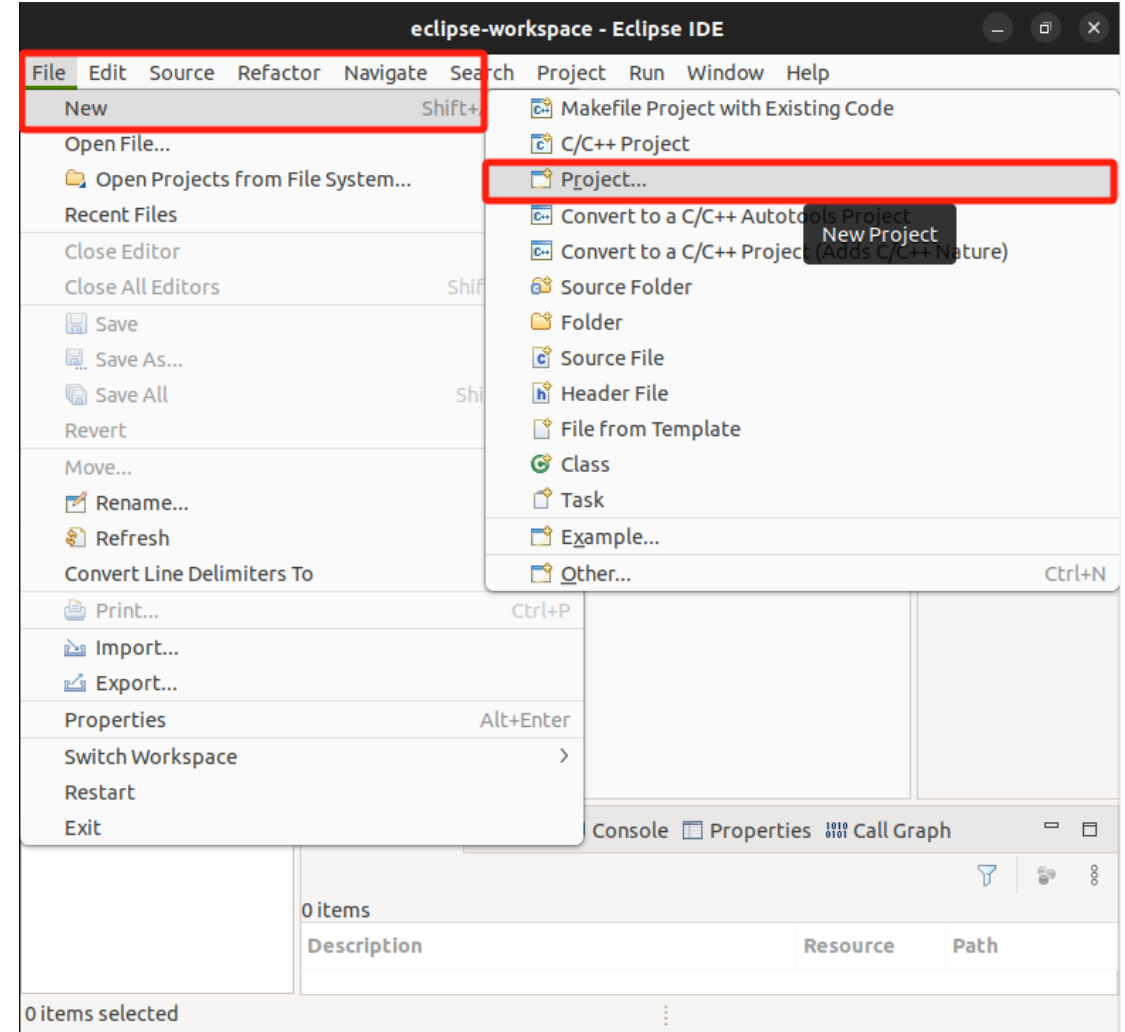
# | Launching Eclipse

- From Application Launcher, find the Eclipse and run it.



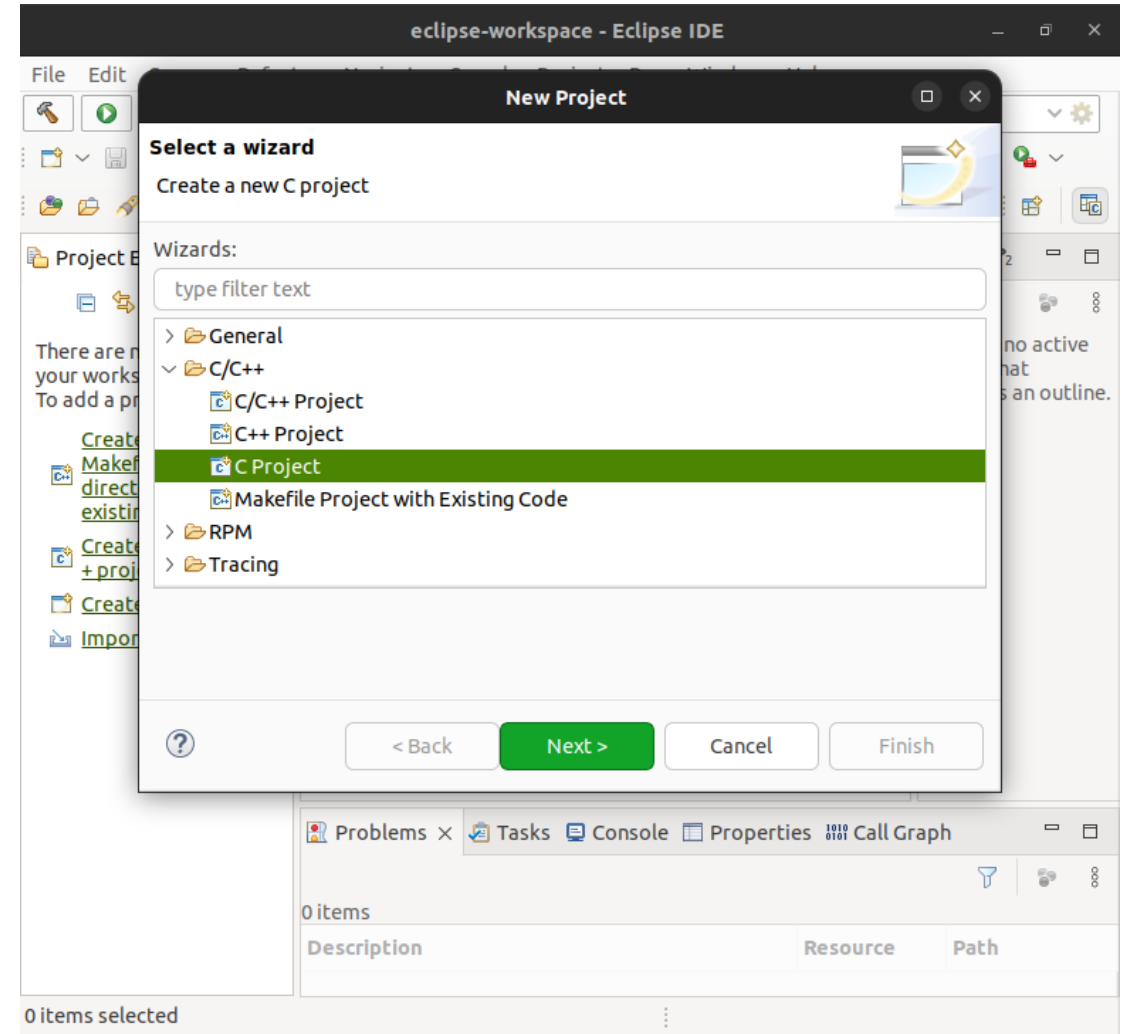
# | Creating Project

- Create a *New Project* from *File* menu  
→File →New →Project



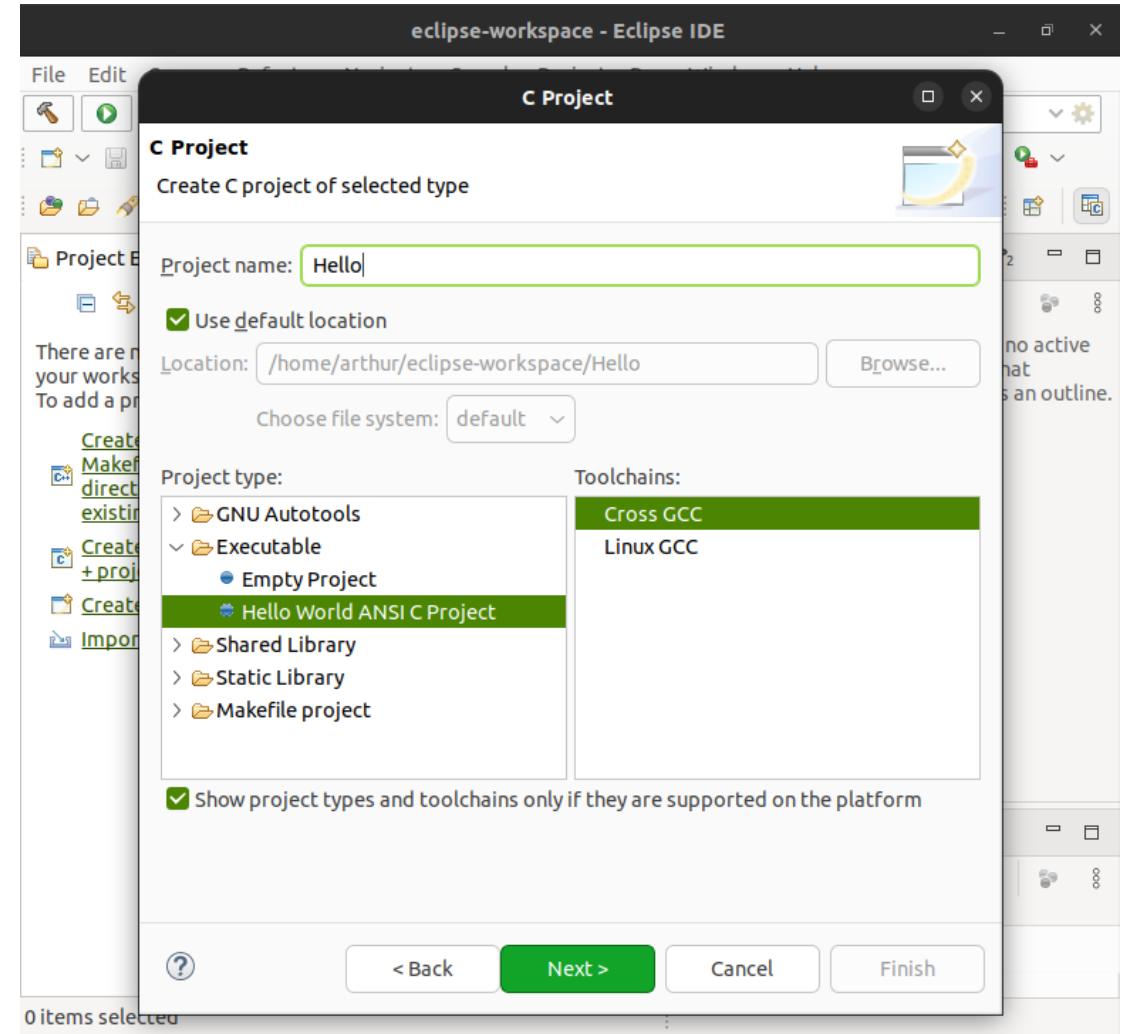
# | Creating Project

- Choose the *C Project*
- Then click the *Next* to forward



# | Creating Project

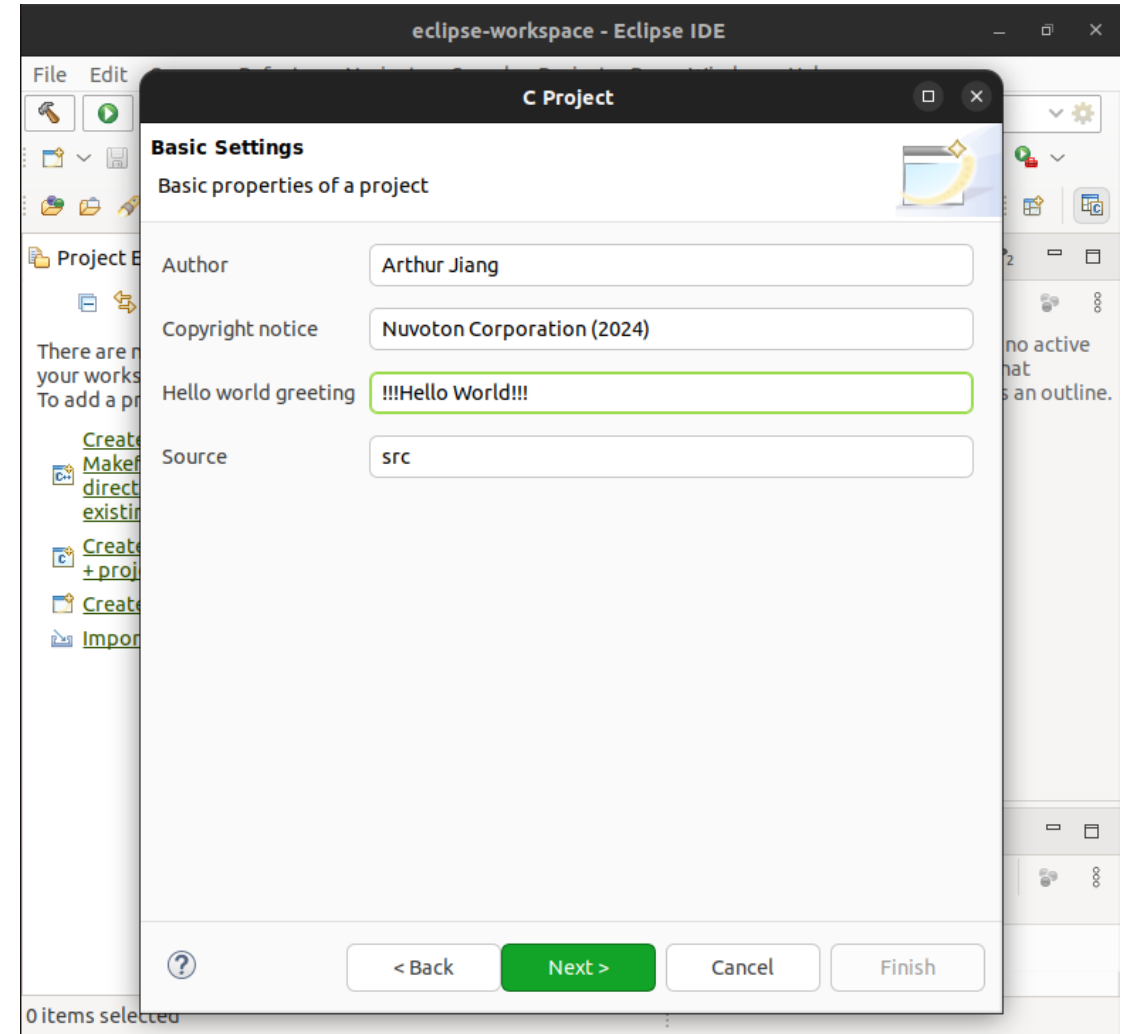
- Give a name *Hello* to the new project
- Choose the *Project type* to *Executable*
- Select *Toolchains* as *Cross GCC*
- Click the *Next* to proceed





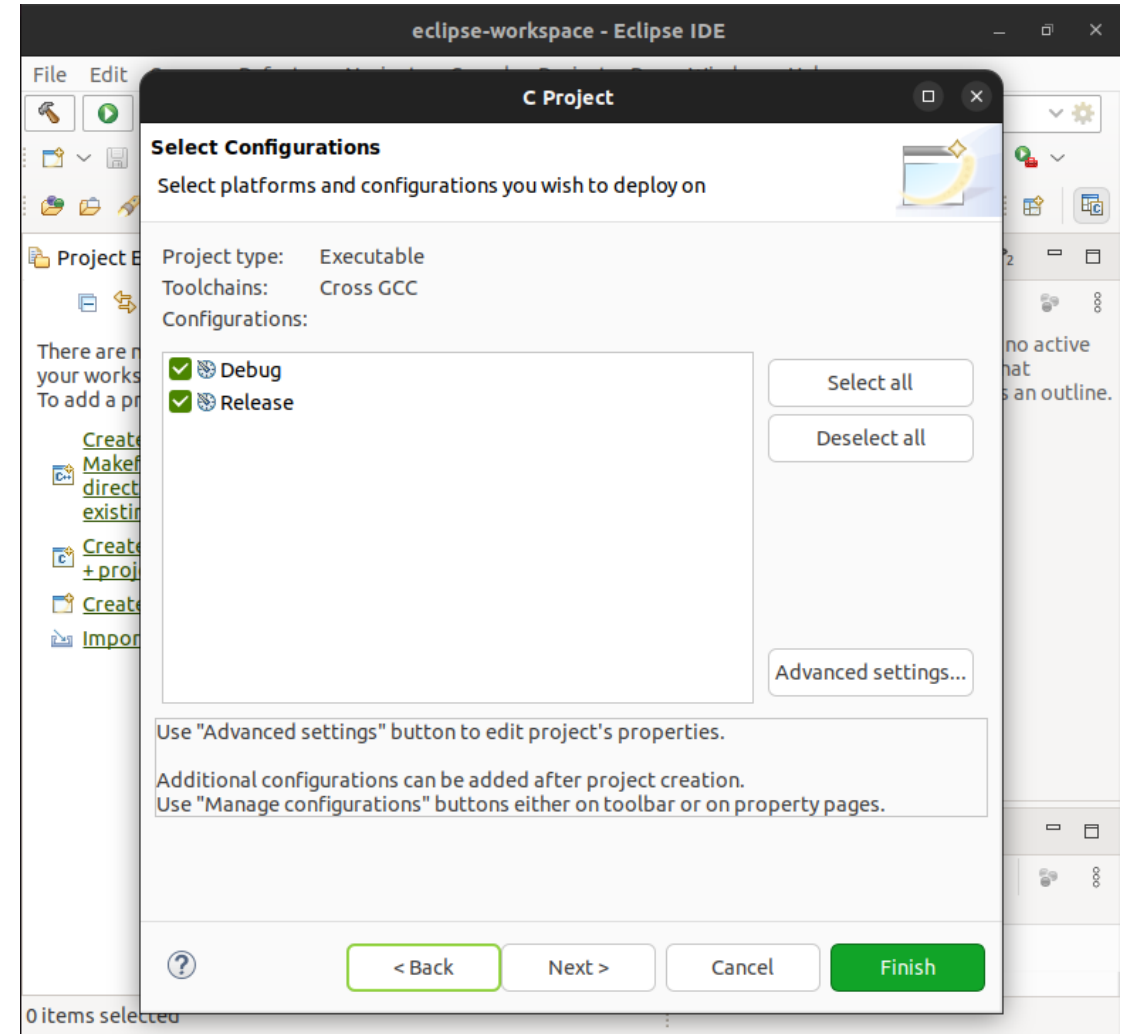
# | Creating Project

- Fill the *Basic Settings* with author, copyright notice and greeting.



# | Creating Project

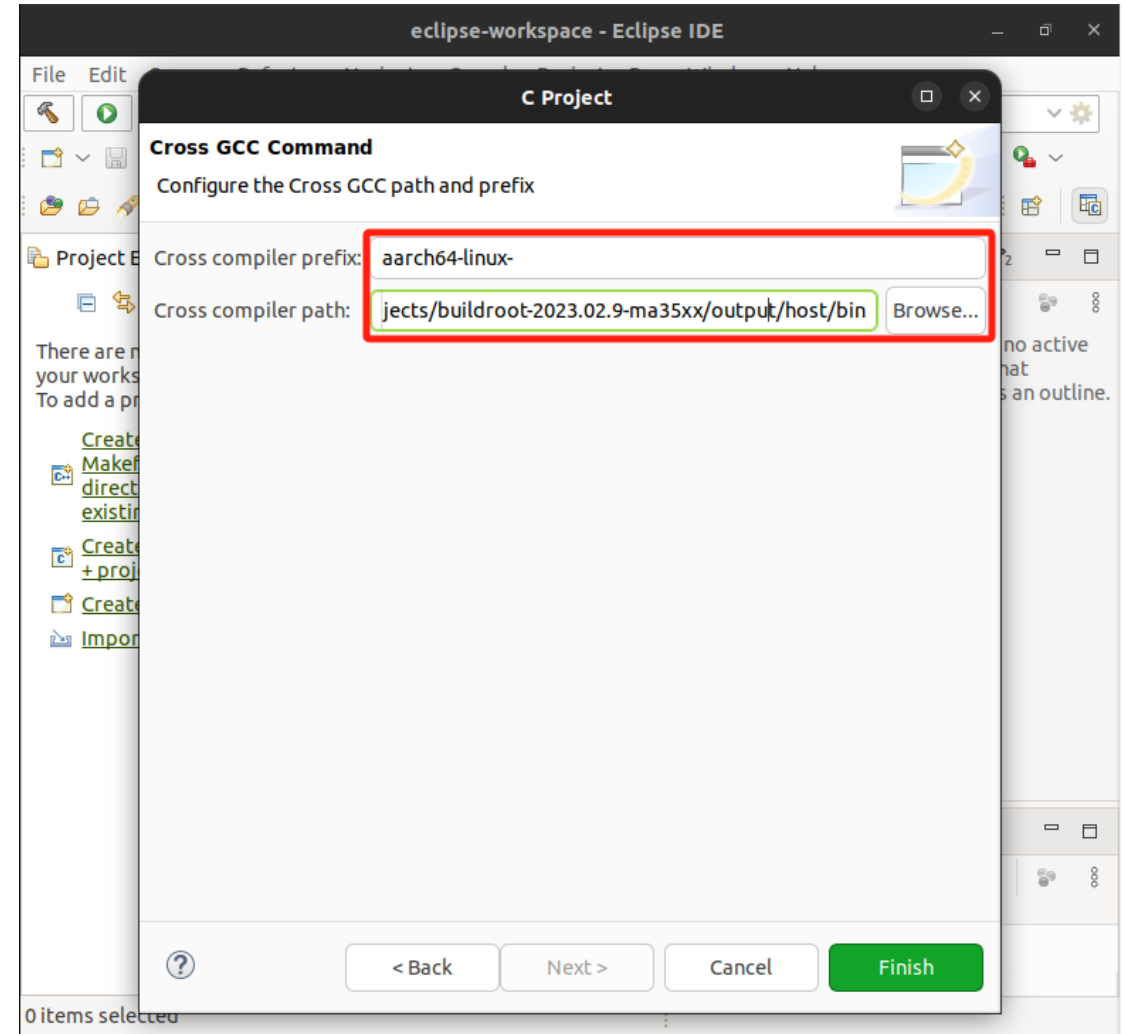
- Select both configurations *Debug* and *Release*



# | Creating Project

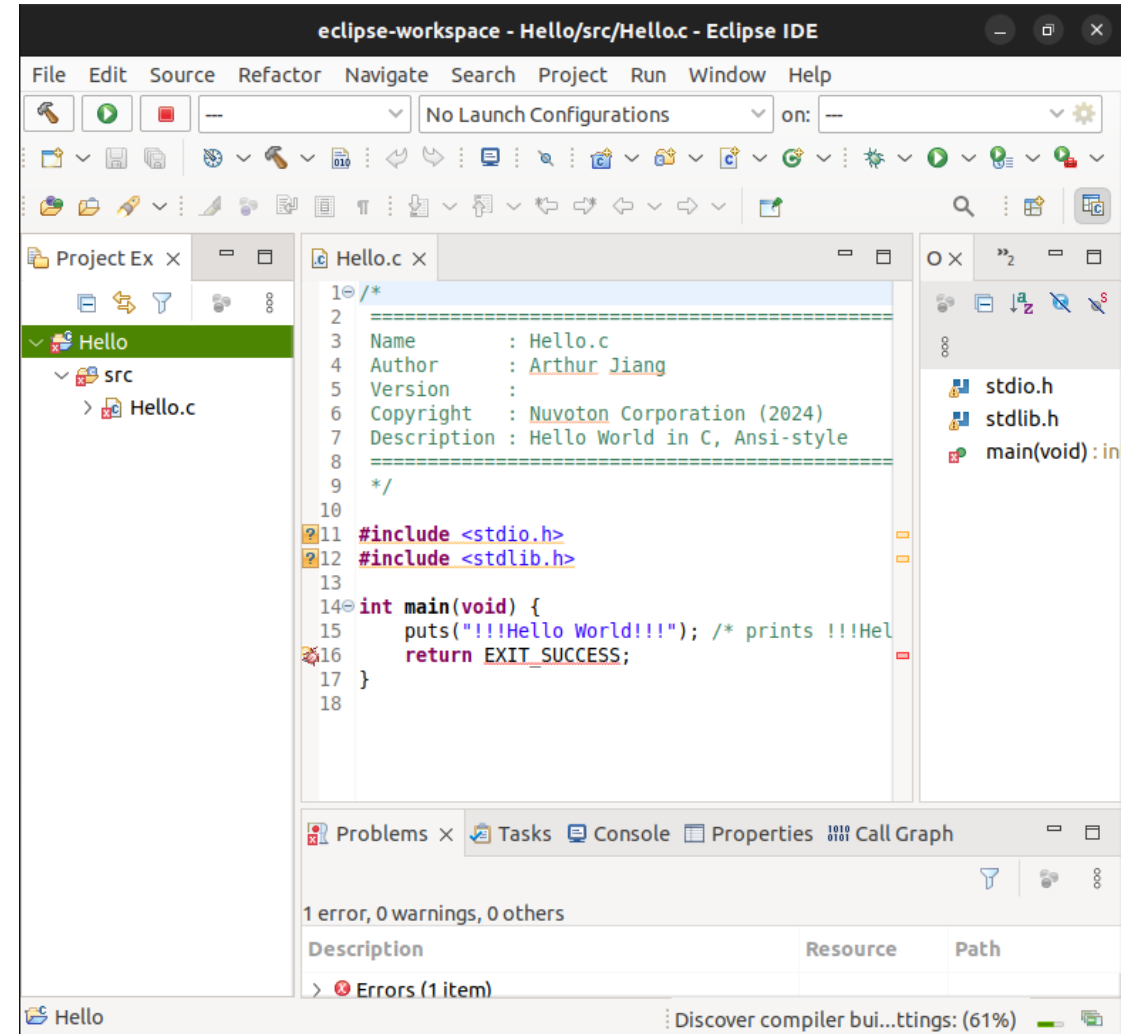
- Fill **Cross Compiler Prefix** with *aarch64-linux-*
- Fill **Cross Compiler Path** with *\${BR2\_DIR}/output/host/bin*

NOTE: ***\${BR2\_DIR}*** is the root directory of Buildroot. Do not use environment variable ***\${BR2\_DIR}*** here, use the actual path of Buildroot instead.



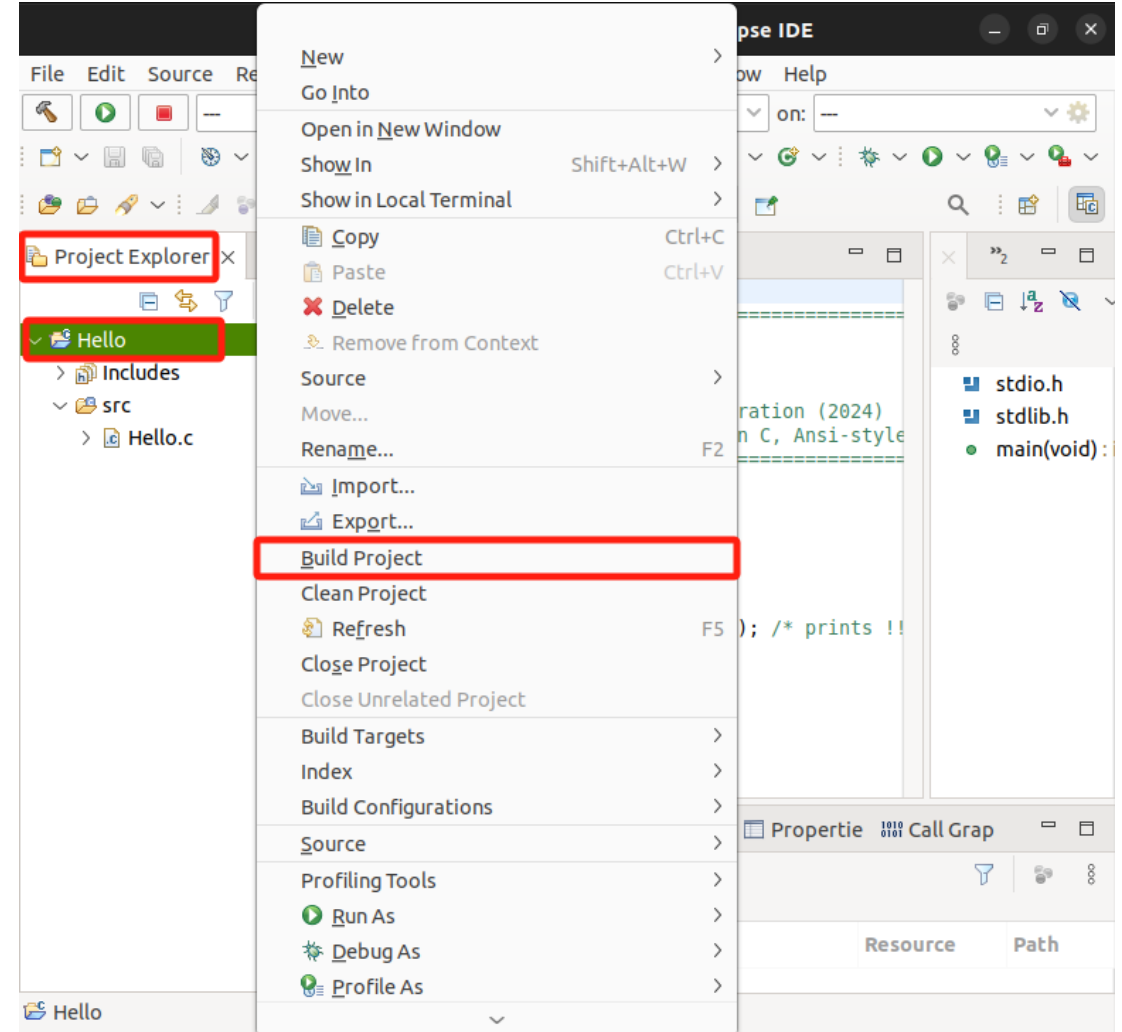
# Building Project

- Before debugging the remote target, project must be built in **Debug** mode.



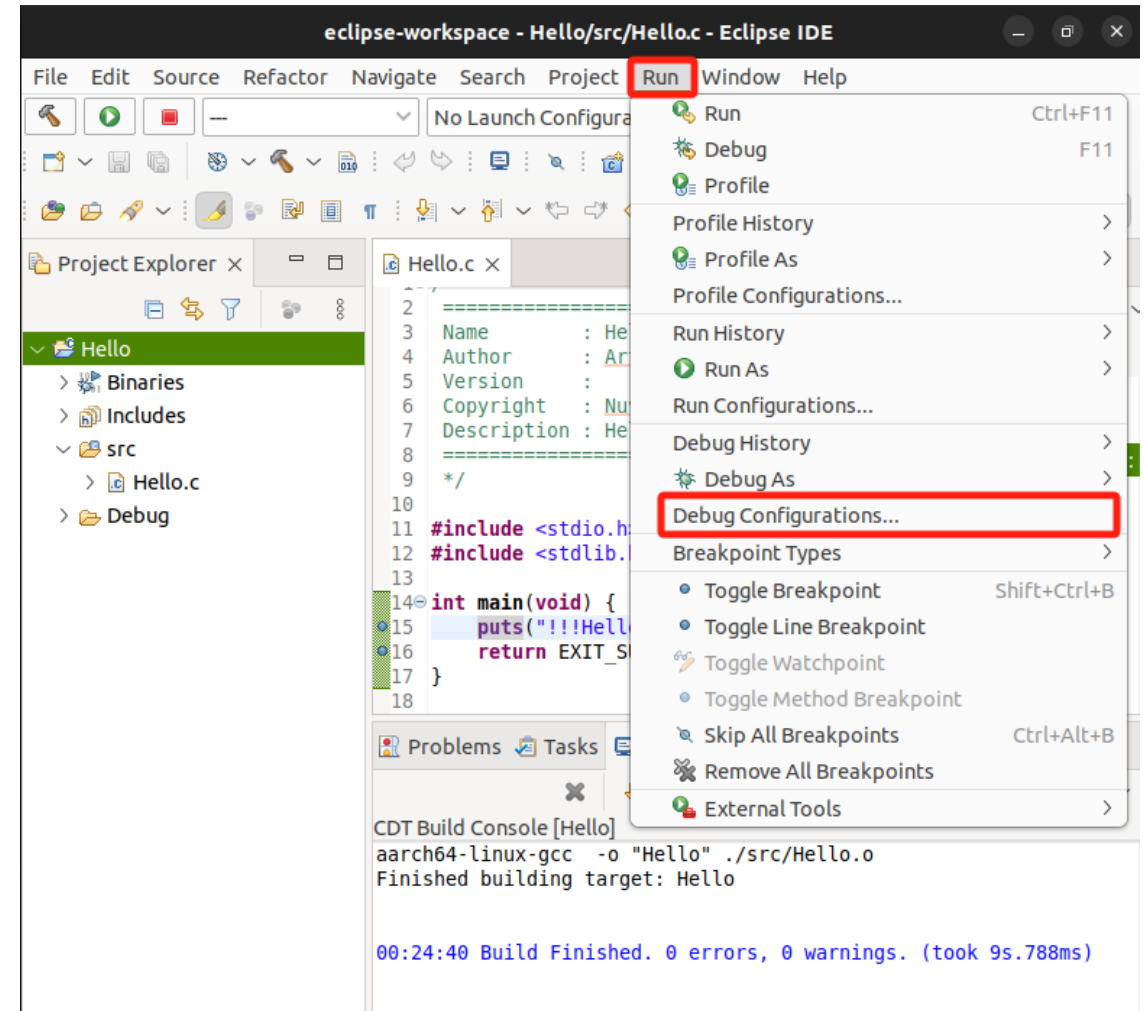
# | Building Project

- Browse the **Project Explorer**, select the project, right click to pop up the context menu, choose **Build Project** to build **Debug** executable target.



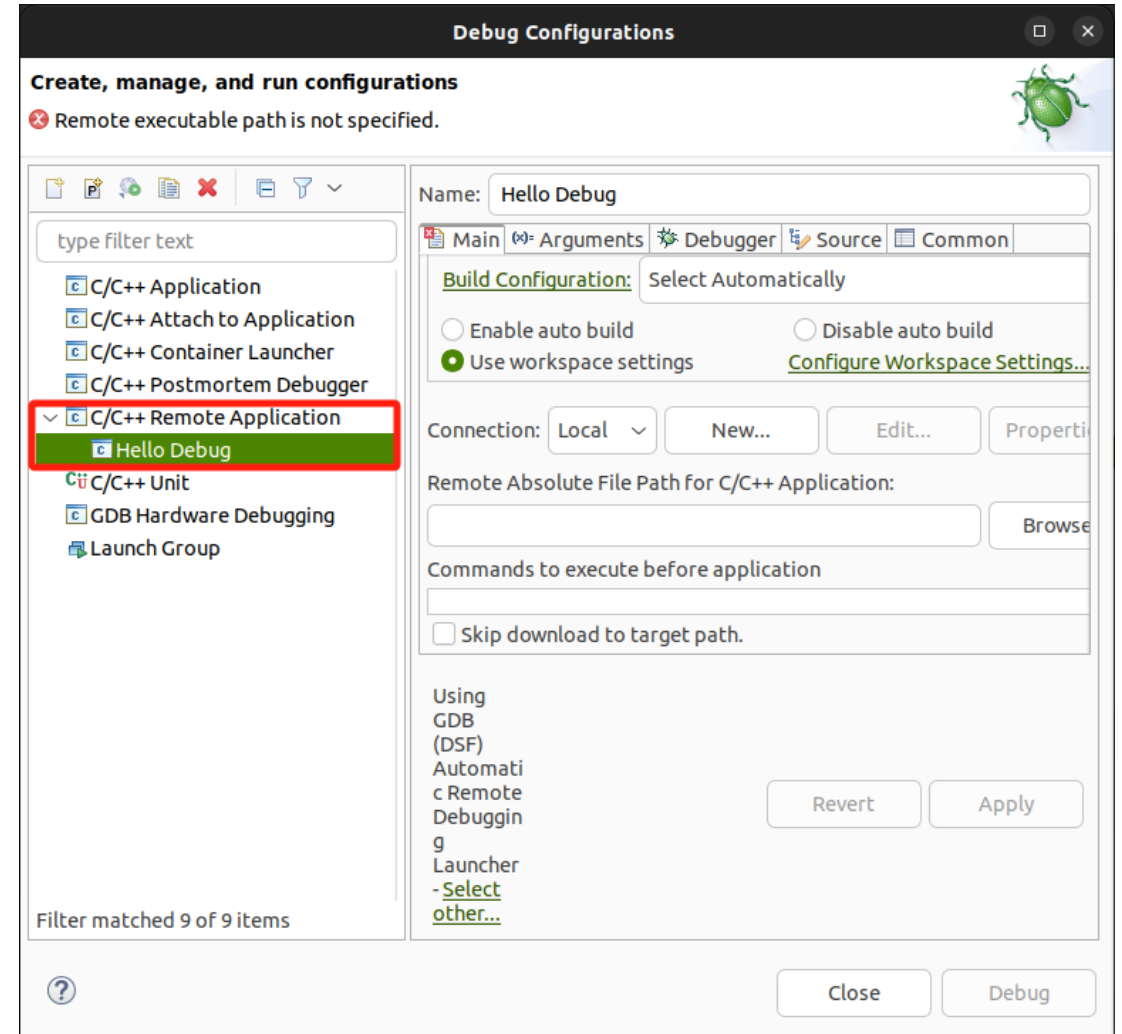
# | Configuring Debug

- Click the **Run** item in menu bar
- Select the **Debug Configurations**



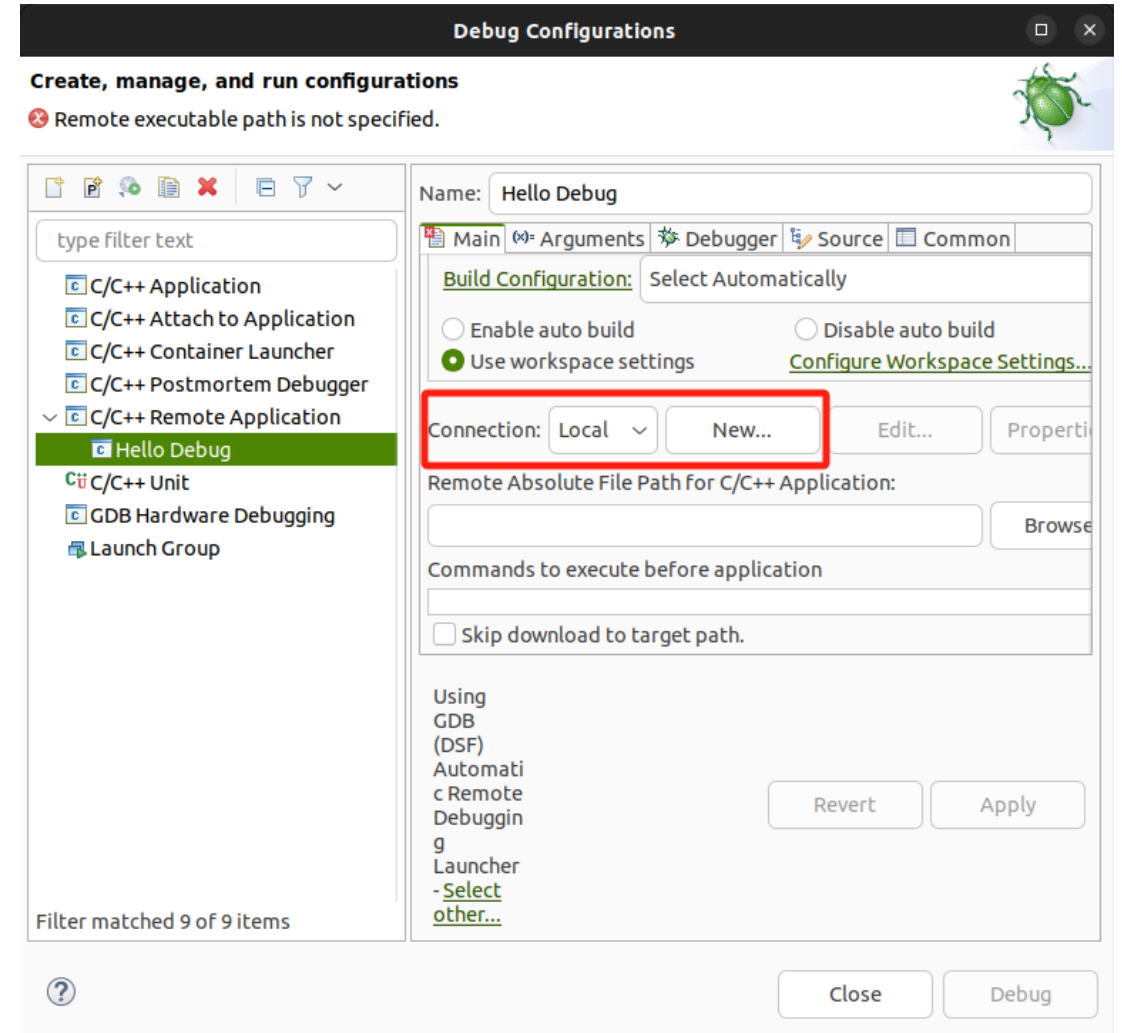
# Configuring Debug

- Double click the **C/C++ Remote Application** to create a remote debugging configuration



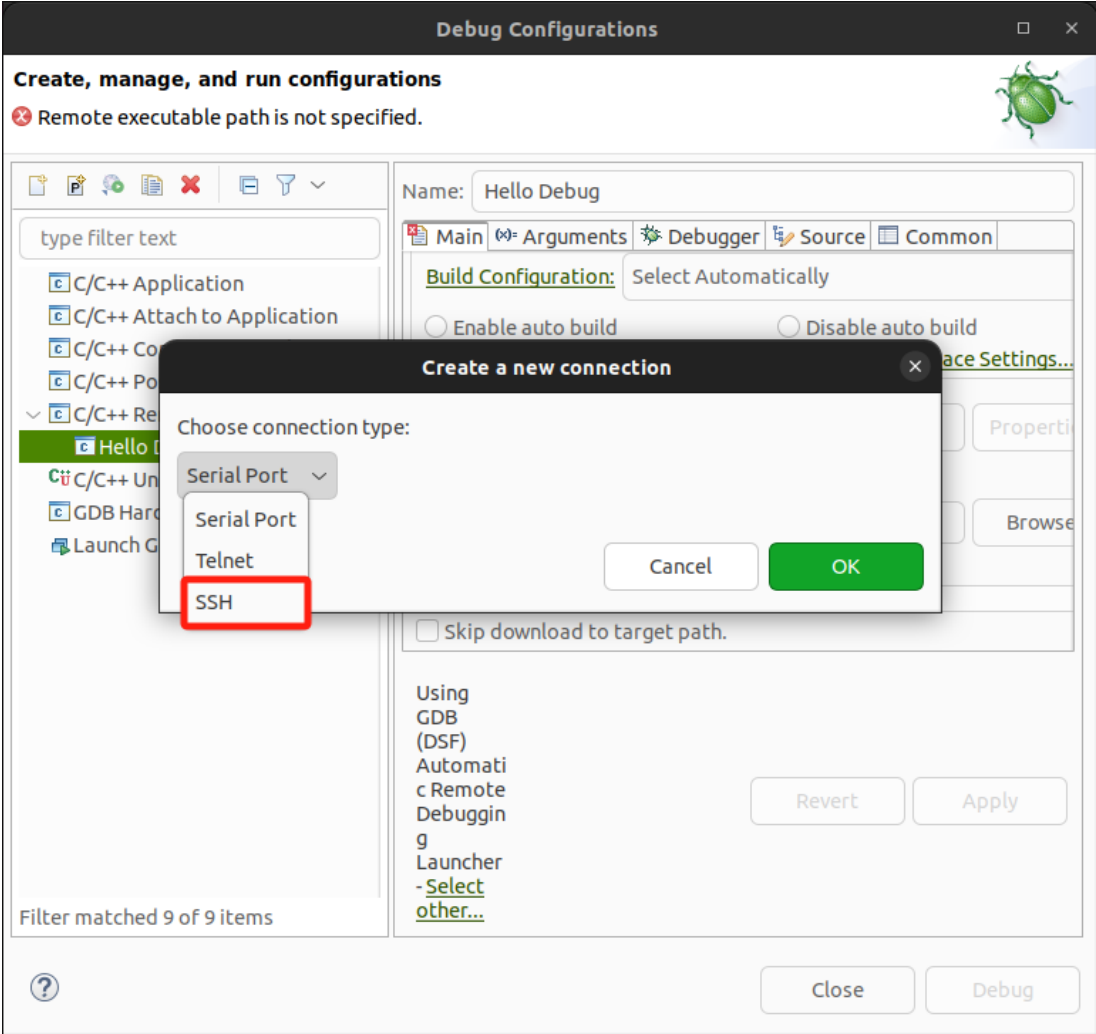
# Configuring Debug

- Click the **New** button to create a SSH connection



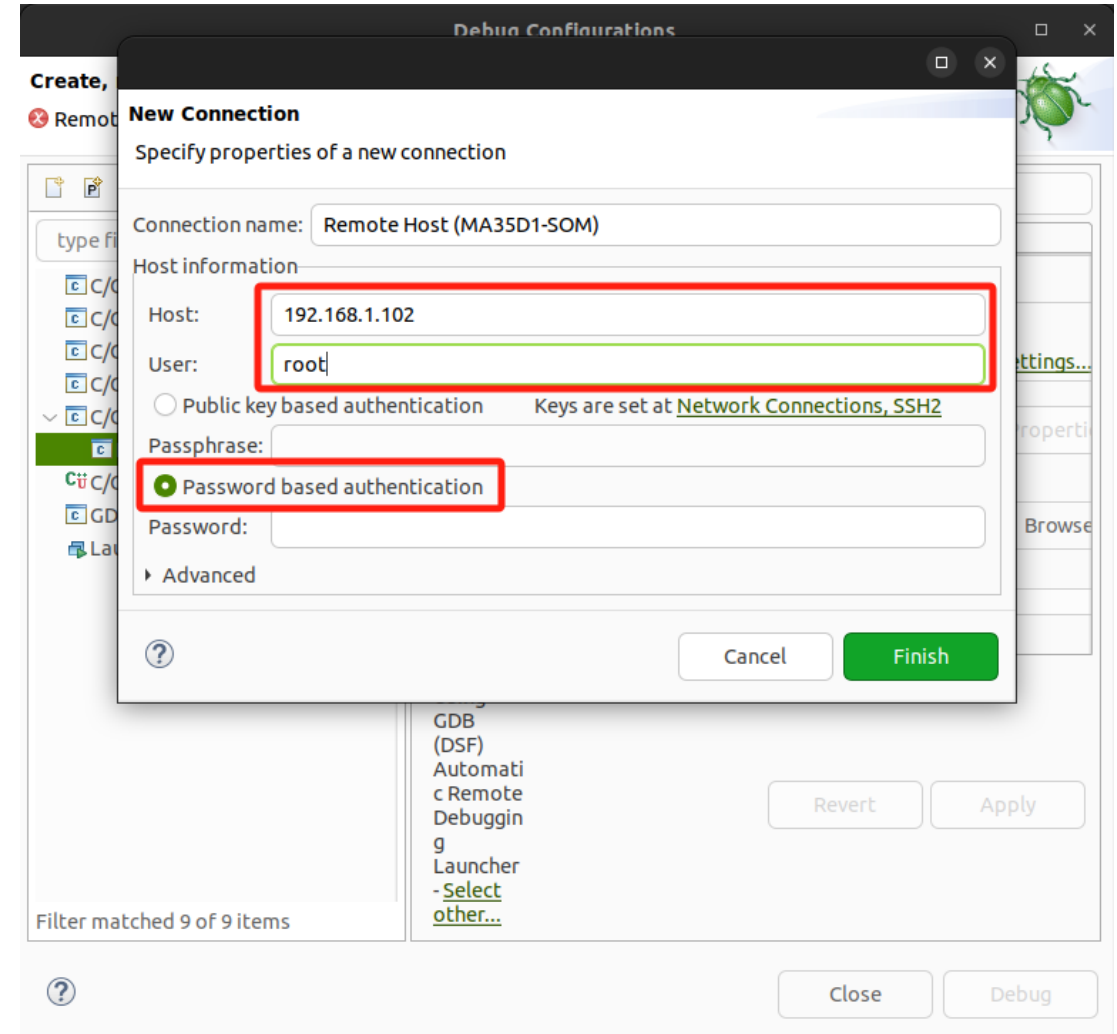


- Choose the connection type: **SSH**



# Configuring Debug

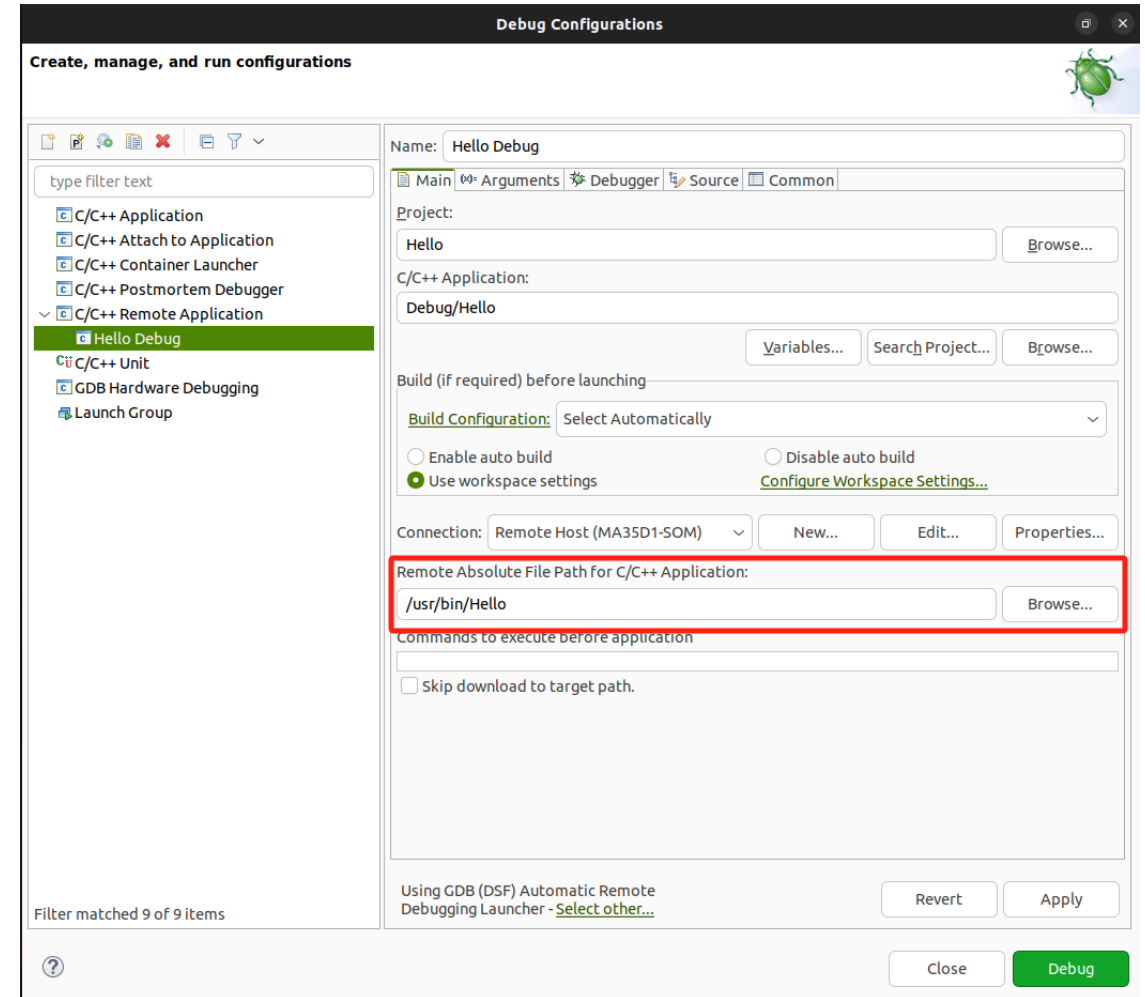
- Fill **Host** with actual IP address (*192.168.1.102*) of remote target board
- Fill the **User** with *root*
- Choose the **Password based authentication**
- Name the Connection to *Remote Host (MA35D1-SOM)*



# | Configuring Debug

- Click Browse to set **Remote Absolute File Path for C/C++ Application**.

**NOTE:** This can test whether the *SSH connection* is lost.

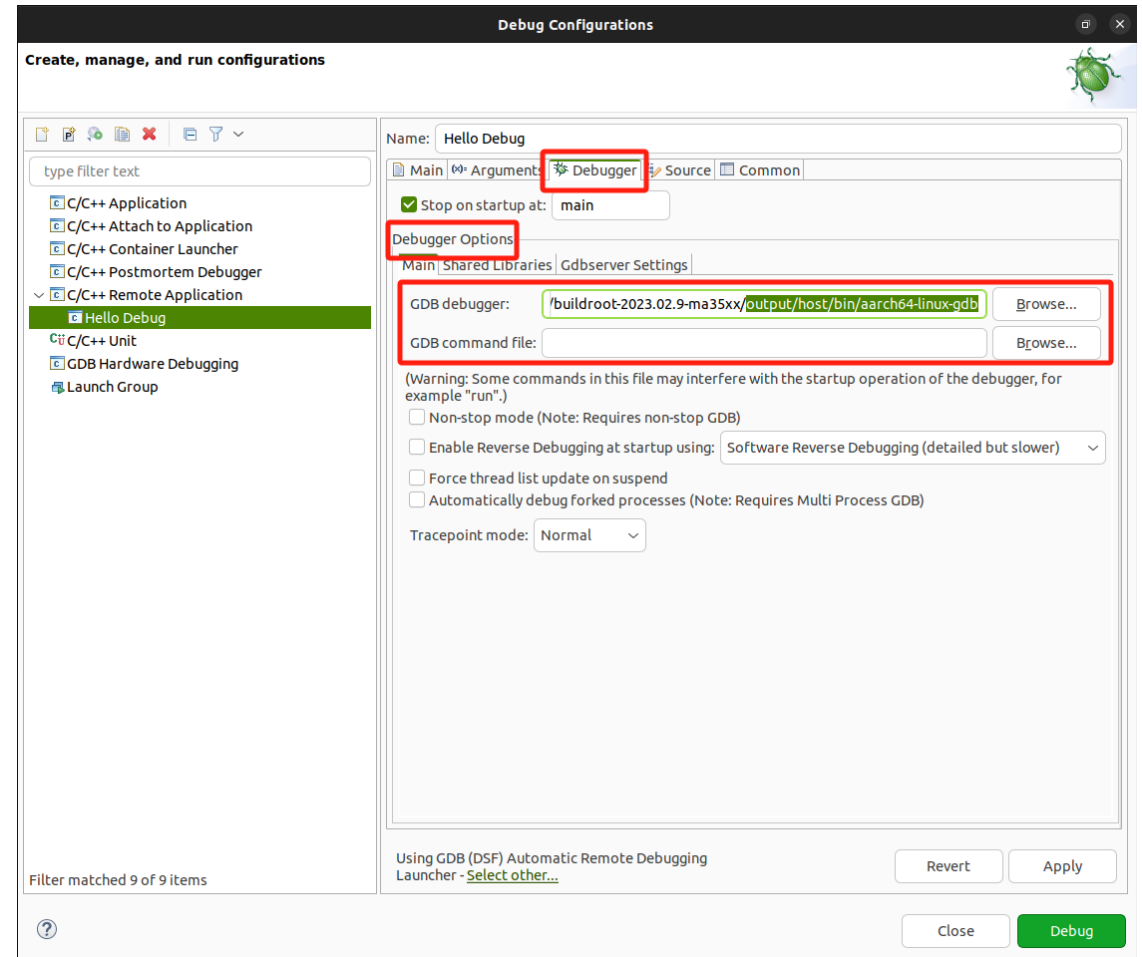


# | Configuring Debug

- In Tab page **Debugger**, under **Debugger Options**, set **GDB debugger** to `${BR2_DIR}/output/host/bin/aarch64-linux-gdb`

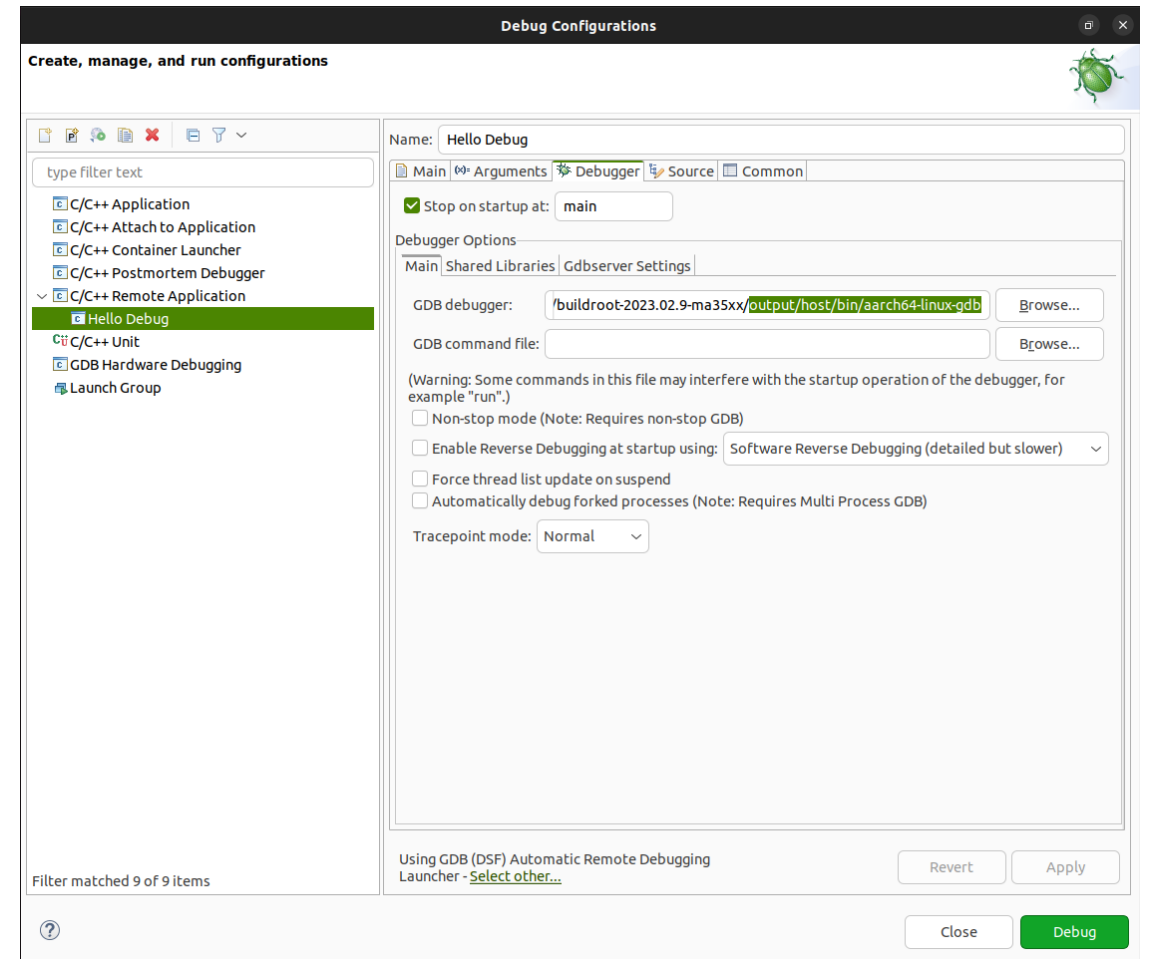
NOTE: `${BR2_DIR}` is the root directory of Buildroot. Do not use `${BR2_DIR}` here, use the actual path of Buildroot instead.

- Leave the **GDB command file** *blank*



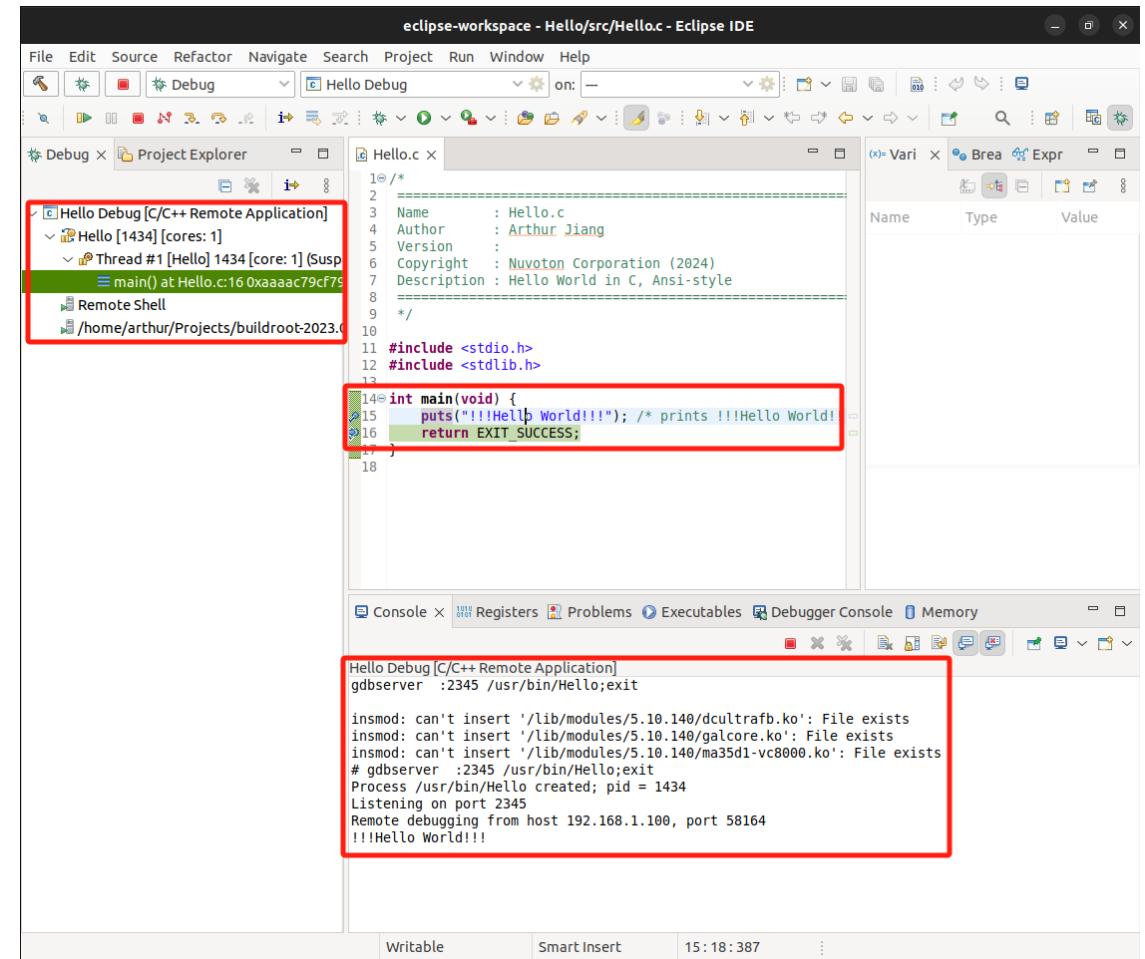
# | Beginning Debugging

- Click **Debug** to begin debugging



# Debugging Target

- Double click a line of source code to toggle breakpoint



*Joy of innovation*  
**nuvoTon**

谢谢

謝謝

Děkuji

Bedankt

Thank you

Kiitos

Merci

Danke

Grazie

ありがとう

감사합니다

Dziękujemy

Obrigado

Спасибо

Gracias

Teşekkür ederim

Cảm ơn