FreeRTOS Simulator Test Procedure for Windows using Eclipse.

STEP 1: First install the MinGW

MinGW is a software package which provides GNU tools for windows such as C/C++ compiler, linker, etc

Go here: http://www.mingw.org/category/wiki/download

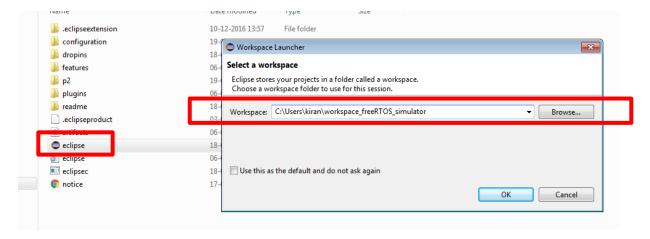
Then click on "Download installer"



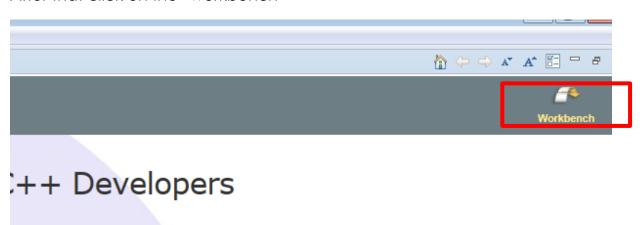
After downloading you will be having the file "**mingw-get-setup.exe**" which you need to install.

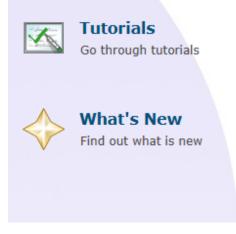
STEP 2: Create an Eclipse project for ANSI 'C' Project

First double click on "Eclipse" icon and give a name for the workspace as shown below



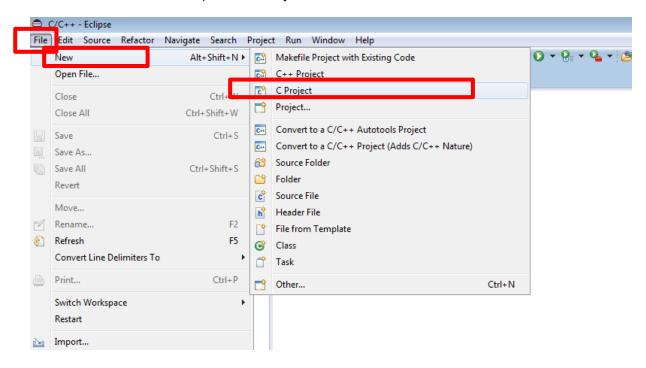
After that click on the "workbench"





Great! Now we are in the workbench of our workspace

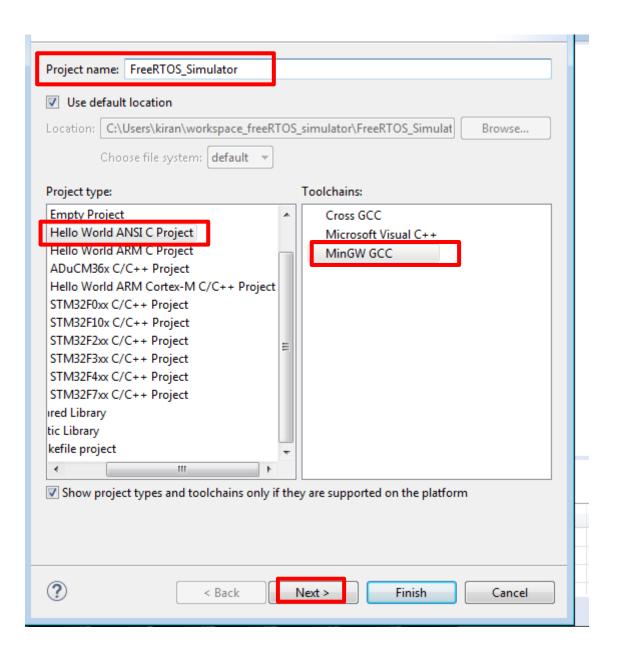
Now let's create a simple 'C' Project.



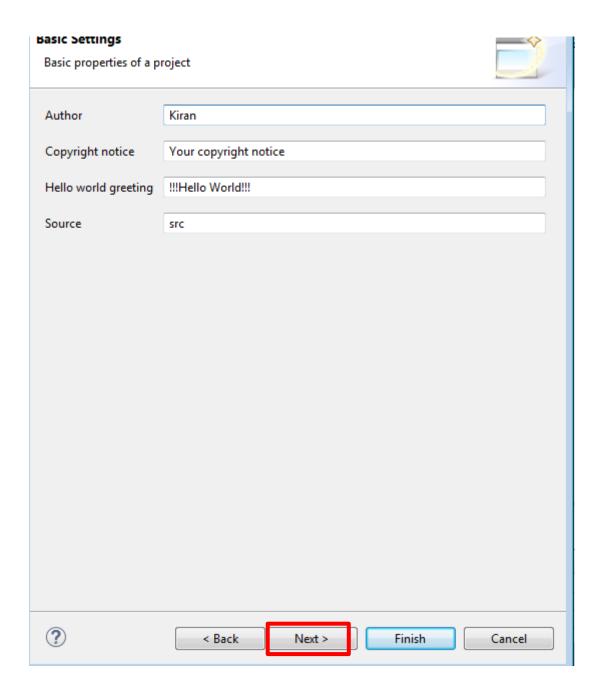
Give a name for your project

And select other options as shown in the below picture

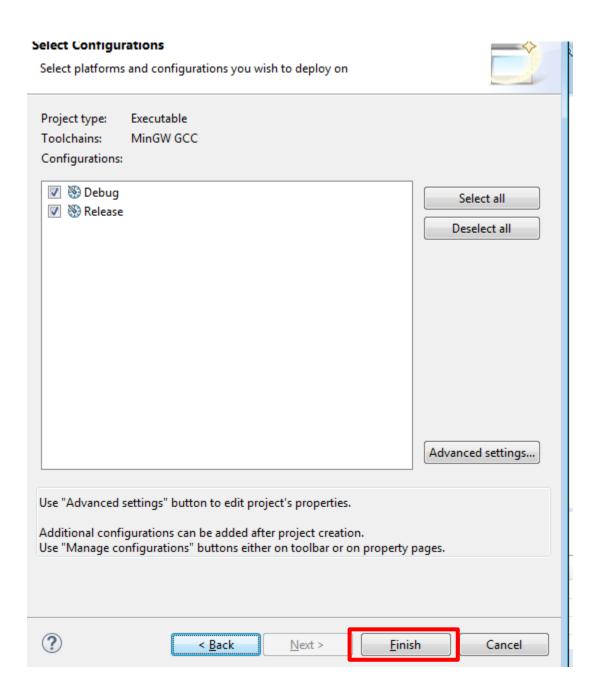
Then click "Next"



Fill up the below details if you want, then click "Next"



Now click "finish"



Great! Now you created ANSI C project.

```
File Edit Source Refactor Navigate Search Project Run Window Help

Project Explorer 
FineRTOS_Simulator.c

FreeRTOS_Simulator.c

Fre
```

STEP3: Add FreeRTOS kernel source with windows port.

Now, in this step let's add a freeRTOS kernel source to our project. So far you have seen that we have to change the **port.c** for different architectures.

port.c is the one which consist of arch specific codes. Now ,since we are executing freeRTOS on windows machine itself, we have to add **port.c** which is specific to windows machine. port.c is given by freeRTOS itself.

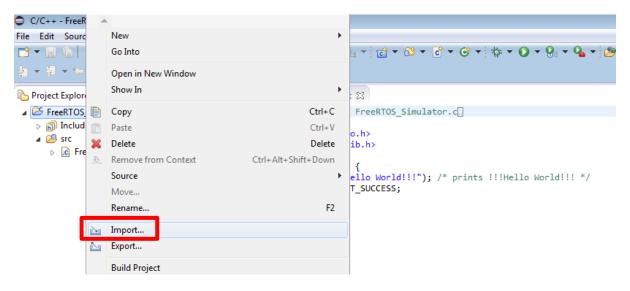
Now, first let's add the freeRTOS kernel source to our project.

Go to the path,

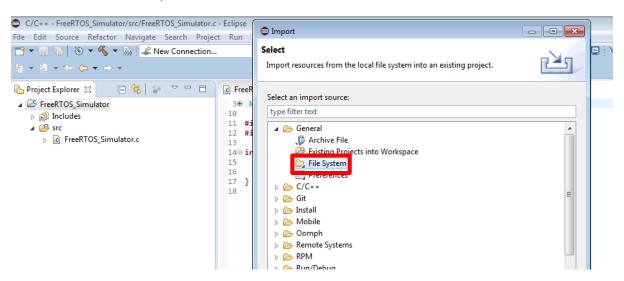
Resources_RTOS\FreeRTOS+Simulator

First let's add the freeRTOS kernel source.

Right click on your project and then click "import"



Now, click on "Filesystem" then Click "Next"



Now, give the path where you have stored "Resources_RTOS" folder

<Your_path>\Resources_RTOS\FreeRTOS+Simulator

And select "FreeRTOSv9.0.0" which is nothing but freeRTOS kernel source

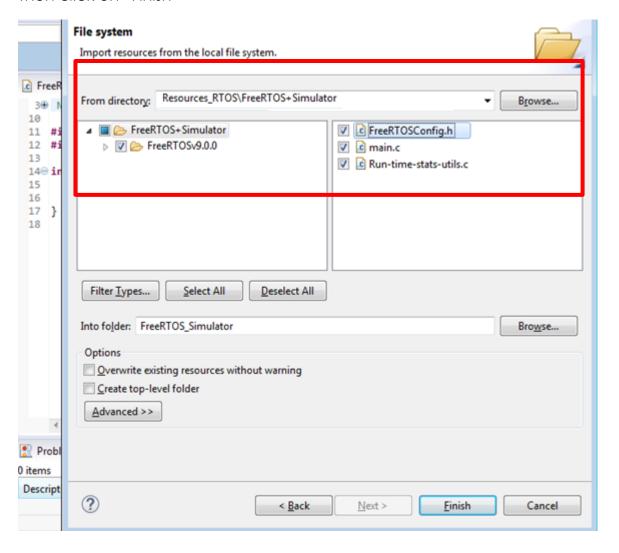
FreeRTOSConfig.h, which is nothing but the configuration file for the freeRTOS

main.c: a sample freeRTOS application to test our simulator

Run-time-stats-utils.c: this is a source file which gives lots of stats related functions for windows. Nothing to do with RTOS.

Select as below

Then click on "Finish"



Great now you have added freertos kernel source to the project.

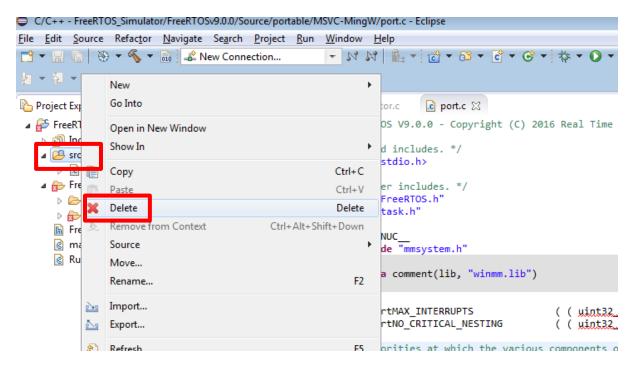
Please note that, this is the port.c which is specific to windows.

```
The second of th
                                                                                                                                          9 ▼ ₩ + ★ → ▼
Project Explorer 🖂
                                                             \neg \neg \sqcap
                                                                                                                           c FreeRTOS_Simulator.c
                                                                                                                                                                                          © port.c ⊠
  FreeRTOS_Simulator
                                                                                                                                                      FreeRTOS V9.0.0 - Copyright (C) 2016 Real Time
         ▶ 👘 Includes
                                                                                                                                 70 /* Standard includes. */
         Src
                                                                                                                                 71 #include <stdio.h>
         72
                License
                                                                                                                                 73 /* Scheduler includes. */
                74 #include "FreeRTOS.h"
                                                                                                                                 75 #include "task.h"
                       include
                                                                                                                                 76
                        77 #ifdef GNUC
                               78
                                                                                                                                                     #include "mmsystem.h"

→ MSVC-MingW

                                                                                                                                 79
                                                                                                                                                      #pragma comment(lib, "winmm.lib")
                                                                                                                                 80
                                          g port.c
                                                                                                                                 81
                                             局 portmacro.h
                                                                                                                                 82
                              @ croutine.c
                                                                                                                          🍇 83 #define portMAX_INTERRUPTS
                                                                                                                                                                                                                                                       ( ( wint32
                              @ event_groups.c
                                                                                                                                84 #define portNO_CRITICAL_NESTING
                                                                                                                                                                                                                                                       ( ( uint3;
                              @ list.c
                                                                                                                                 85
                                                                                                                                 86⊖ /* The priorities at which the various components
                              @ queue.c
                                                                                                                                 87 Priorities are higher when a soak test is performe
                              readme.txt
                                                                                                                                 88 Windows interfering with the timing. */
                              @ tasks.c
                                                                                                                                 89 #define portSOAK_TEST
                              @ timers.c
                                                                                                                                           #ifndef portSOAK_TEST
                                                                                                                                                      #define portDELETE_SELF_THREAD PRIORITY
               FreeRTOSConfig.h
               @ main.c
               Run-time-stats-utils.c
```

Now, we don't need this "src" folder so delete it

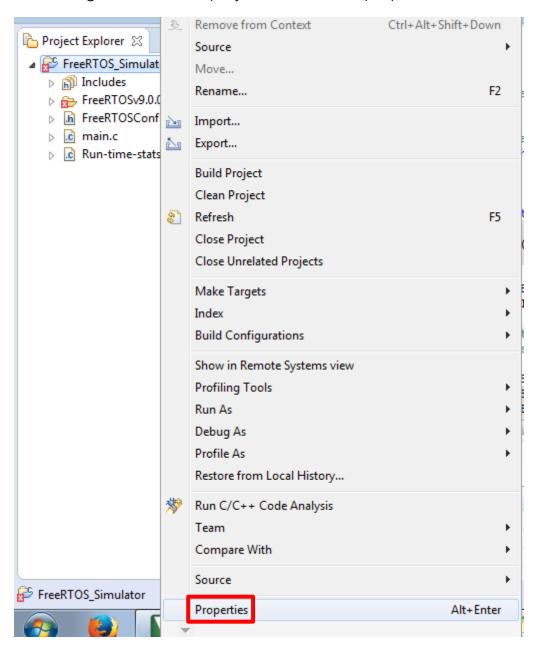


STEP 4: Add include paths to the project

Now let's add all the FreeRTOS header paths to the project, otherwise compile will issue errors.

First lets add the freertos header files path to the project.

For that right click on the project and select "properties"

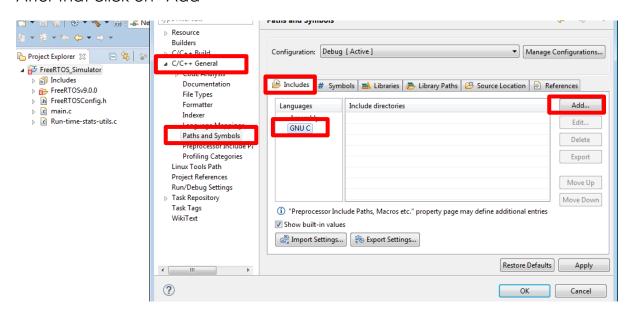


Now, expand "C/C++ General"

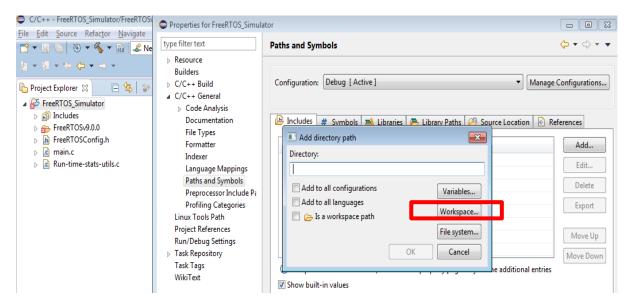
Select "Paths and Symbols"

Then select "GNU C"

After that click on "Add"

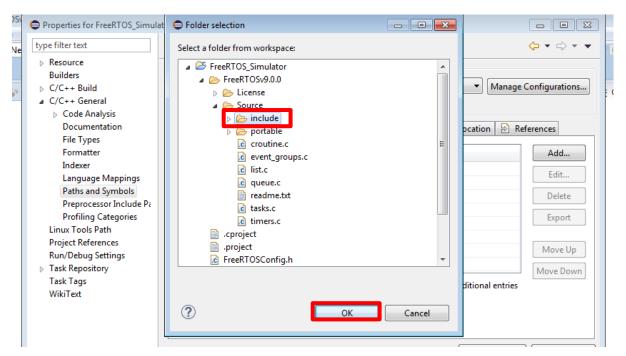


Here, click on workspace



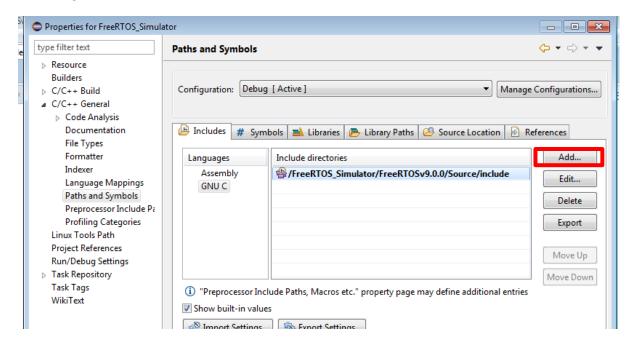
Expand the project and select "include" then click "Ok"

After that again click "ok"

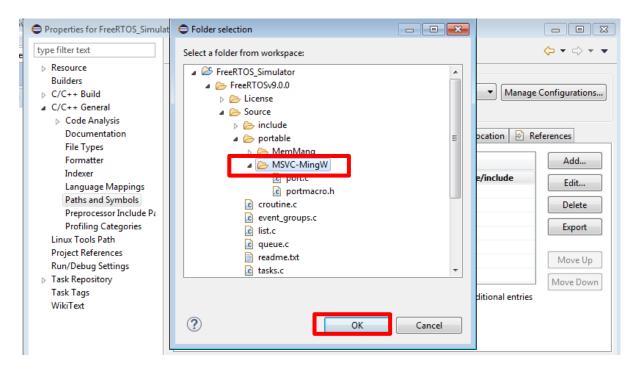


Now, you have added the freeRTOS "include" path

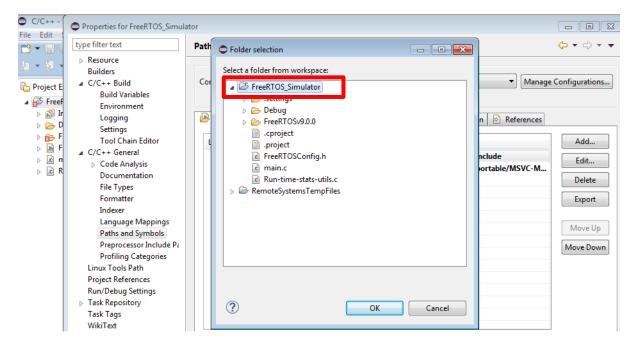
Now again click on "Add" and repeat what we did in last step



This time select the path where we have stored 'port' related header file Then click "OK"

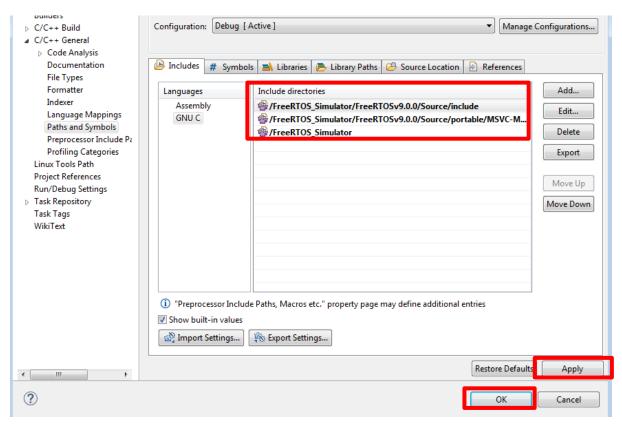


Now, again repeat the step, this time add the path where we have stored the "FreeRTOsconfig.h"



Great! Now we have hopefully added all the required paths to the project.

Now click "Apply" then "OK"



Now you can see in "port.c" it needs **winmm.lib**, that is a windows specific library, lets add that to our project.

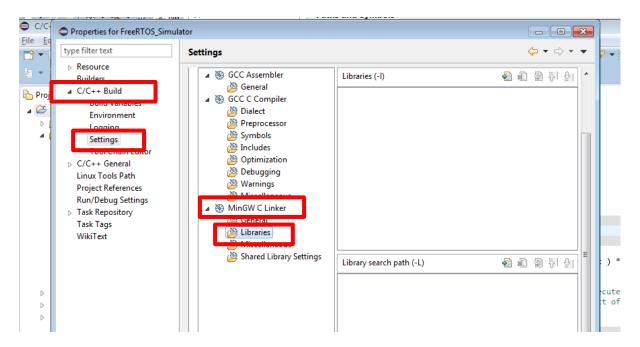
```
C/C++ - FreeRTOS_Simulator/FreeRTOSv9.0.0/Source/portable/MSVC-MingW/port.c - Eclipse
<u>File</u> <u>Edit Source Refactor Navigate Search Project Run Window Help</u>

    The second section  
    The section  
    The second section  
    The section  
    The second section  
    The section  

                                                                                                                                                                    [ 🐉 | 🌶 📮 👉 🔗 😄 🛀 📭 📲 🕶 🚺 🕶 😘 🕶 🐿 🕶 🐿 🕶 😭 🕶 🔀 🔻
  Project Explorer 🖂
                                                                       FreeRTOS V9.0.0 - Copyright (C) 2016 Real Time Engineers Ltd.
                                                                                                                                                      69
             /* Standard includes. */
             71 #include <stdio.h>
                    License
                                                                                                                                                      72
73
                                   license.txt
                                                                                                                                                                /* Scheduler includes. */
                     74 #include "FreeRTOS.h"
75 #include "task.h"
                             > 📂 include
                              > 📂 portable
                                                                                                                                                                            def __GNUC__
#include "mmsystem.h"
                                   croutine.c
                              #pragma comment(lib, "winmm.lib")
                              #define portMAX_INTERRUPTS
#define portNO_CRITICAL_NESTING
                                                                                                                                                                                                                                                                                            ( ( uint32_t ) sizeof( uint32_t ) * 8UL ) /* The ( ( uint32_t ) 0 )
                             readme.txt
            ▶ In FreeRTOSConfig.h
                                                                                                                                                      86\Theta /* The priorities at which the various components of the simulation execute.
```

Right click on the project and select "properties".

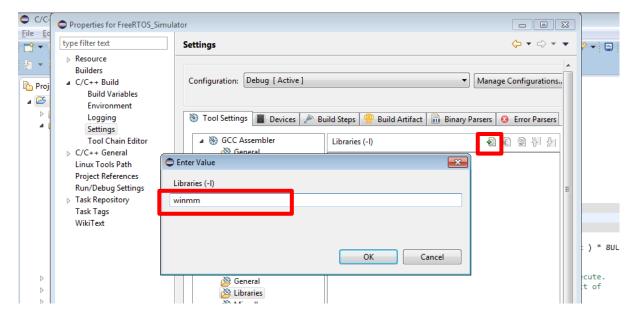
Expand "C/C++ Build", Select "Settings", then select "Libraries" under "MinGW C linker"



In the libraries section click on '+' icon and give the name "winmm"

Then click "OK"

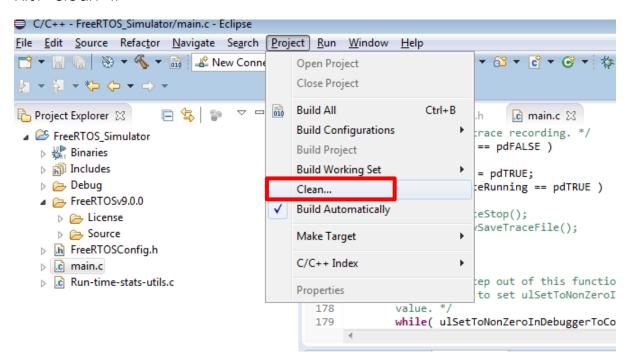
Then "Apply" then "OK"



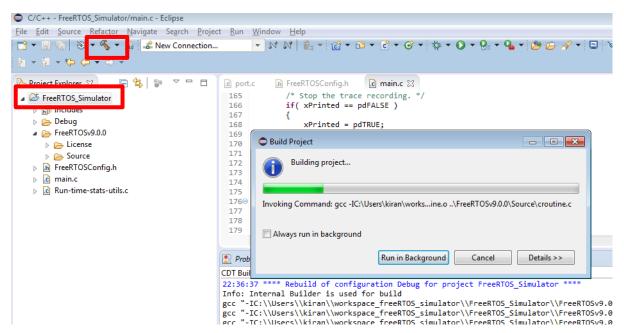
STEP5: Build and Run the project

Now let's build the project

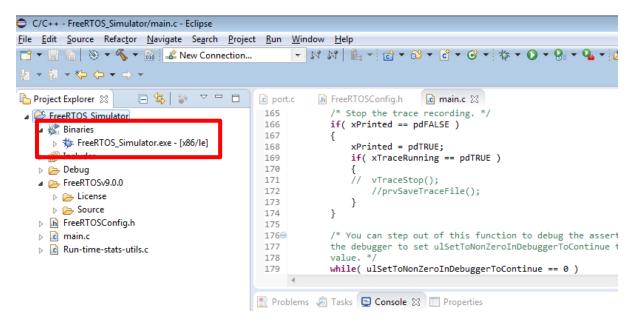
First "clean" it



Then build the project

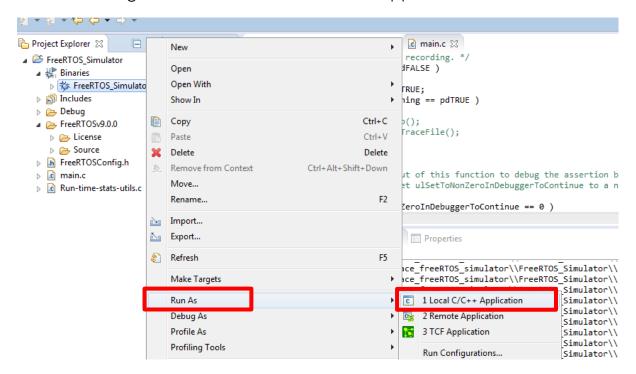


Now, you can see that the .exe file is created.

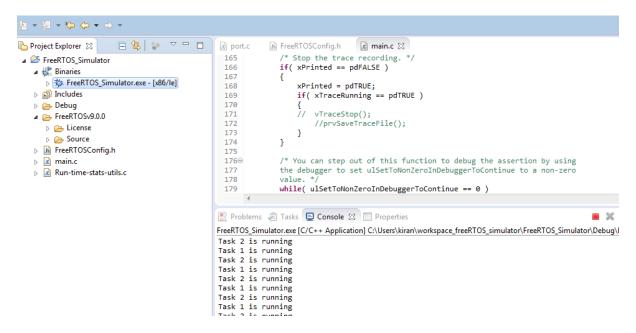


That's nothing but windows executable right? Just execute it!

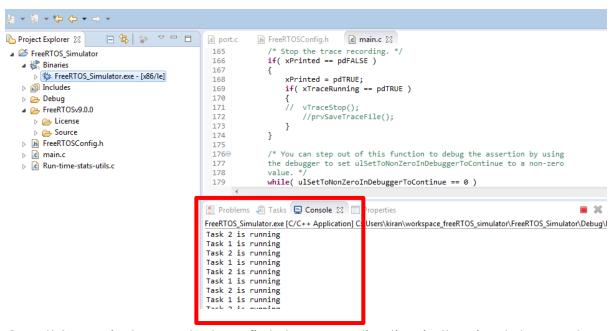
Just select it, right click and run as local c/C++ application



You can see our application is now running,



You can hit the below "red" box to terminate the execution

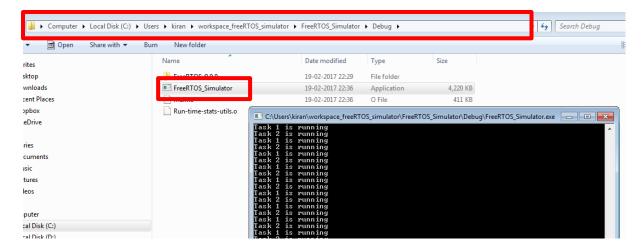


Great! So we just executed our first demo application in the simulator mode.

All you need is just you windows PC and eclipse software, that's it . no hardware is required.

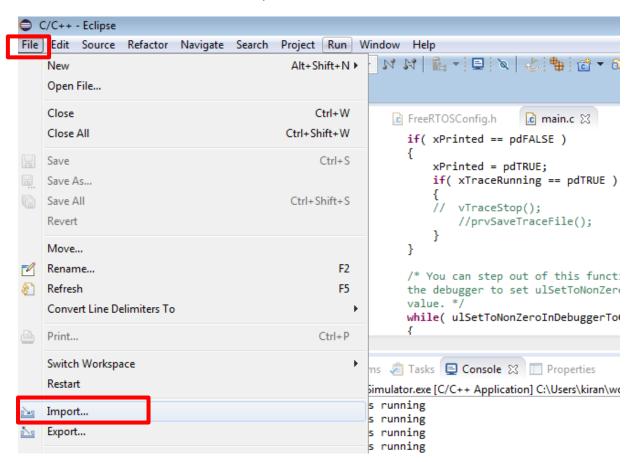
Note 1: some time launching the application as described above may not work properly, in that case you can directly go to the "Debug" folder of the project and execute the ".exe" by just double clicking on it.

See below

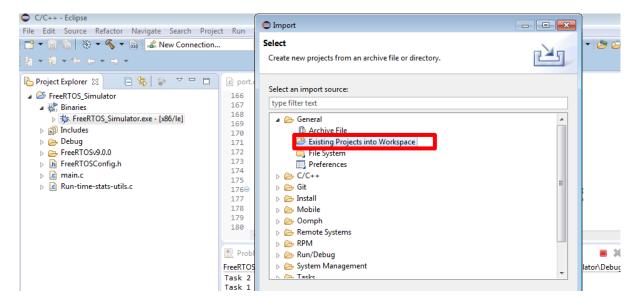


Importing the Course exercises to Current Workspace

Click on "File" Then click on "Import"



Select "Existing Projects into workspace" Then click "Next"



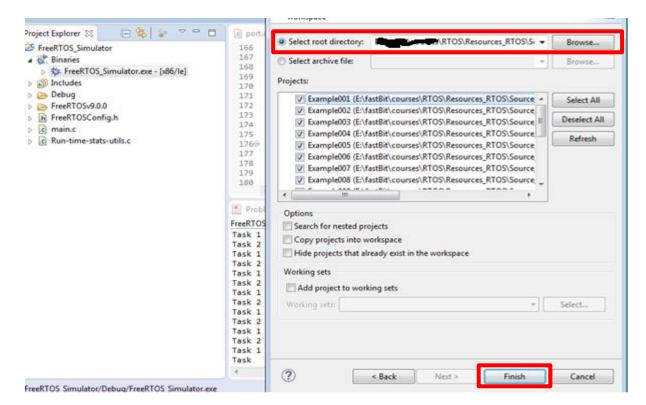
Here, give the path where code exercises for the simulator mode is stored.

The path must be

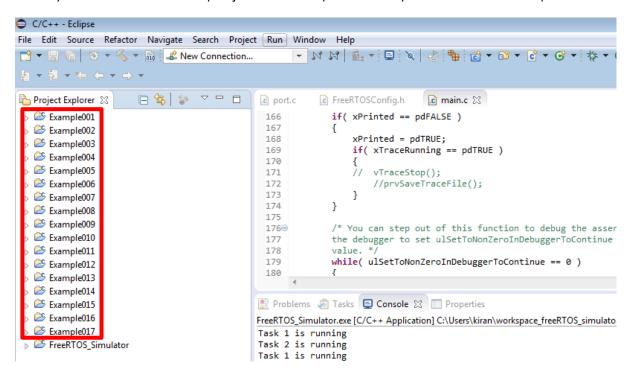
<your path >\ Resources_RTOS\Source_codes\freertos_examples_simulator

After that eclipse will load all the available projects.

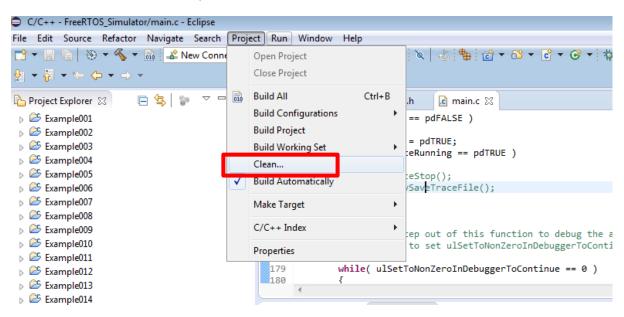
Just click "Finish"



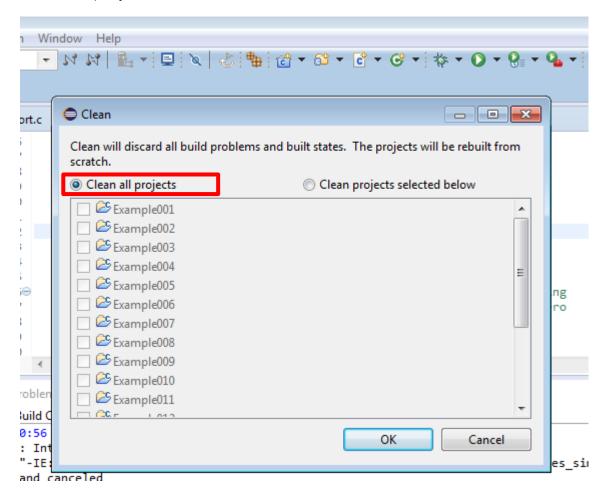
Now you can see all the projects are imported to your current workspace.



First lets clean all the projects.



Clean all projects.



Now, let's build the first project "Example001"

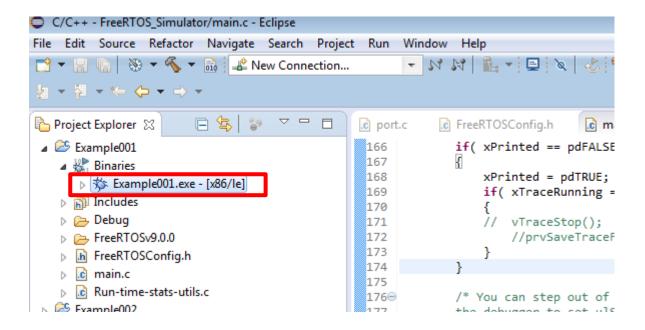
Select it and click on the "hammer" icon

```
C/C++ - FreeRTOS_Simulator/main.c - Eclipse
File Edit Source
                     Navigate Search Project Run
                                             Window Help
□ □ □ ③
                     📆 🕍 New Connection...
                                             \neg \Box
                                                 FreeRTOSConfig.h
Project Explorer 🖂
                                                                   ic main.c ⊠
                                        c port.c

■ Example001

                                        166
                                                   if( xPrinted == pdFALSE )
                                         167
   ⊳ 🛍 Includes
                                         168
                                                       xPrinted = pdTRUE;
   Debug
                                         169
                                                       if( xTraceRunning == pdTRUE )
   170
   ▶ In FreeRTOSConfig.h
                                                          vTraceStop();
                                        171
   172
                                                          //prvSaveTraceFile();
                                        173
   ▶ Run-time-stats-utils.c
                                        174
 175
 Example003
                                                   /* You can step out of this function t
                                        176⊖
```

After building under "Binaries" you will find the .exe created.



Great! Like this you can execute and test all the course exercises on your windows machine.