CE103 Algorithms and Programming I

Introduction to Code Reusability and Automate Testing

Author: Asst. Prof. Dr. Uğur CORUH

Contents

List of Figures

List of Tables

0.1 CE103 Algorithms and Programming I

0.2 Week-4

0.2.0.1 Fall Semester, 2021-2022

Download DOC[1], SLIDE[2], PPTX[3]

[1] ce103-week-4-test.tr.md_doc.pdf
[2] ce103-week-4-test.tr.md_slide.pdf
[3] ce103-week-4-test.tr.md_slide.pptx

0.3 Introduction to Code Reusability and Automated Testing

During this course we will use entry level of shared library development and their tests and test automations. Also we will see TDD(Test Driven Development) approach.

During this course we will use **Windows OS, Eclipse and Visual Studio Commuity Edition** environments for examples.

Each example will include two function

"Hello " printing function with name sayHelloTo(name) and

sum of two variable function for basic, sum = sum(a,b).

This sum function will add a to b and return result to sum variable.

We will locate them in library and use them from a console application, also we will create unit tests for testing their functionalities and return variables

---

0.4 Shared Library Development

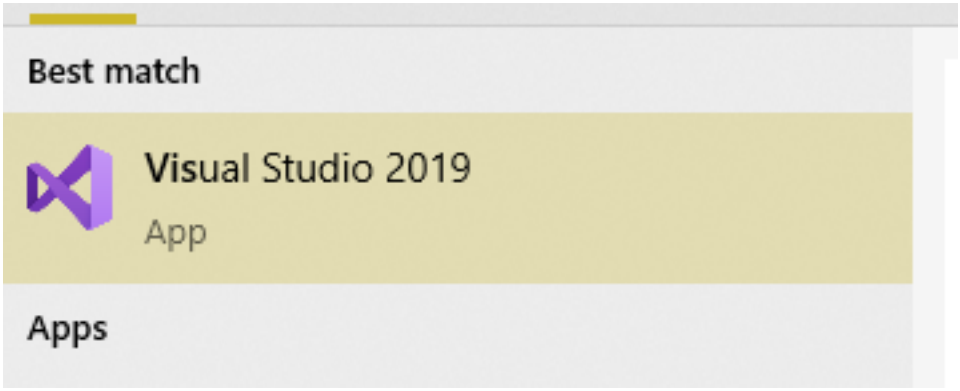
0.4.1 C Programming (Static Library)

0.4.1.1 Visual Studio Community Edition In this sample we will create **c-lib-sample** project that contains library, executable, unit tests and unit test runners.

First of all you install Visual Studio Community Edition from website

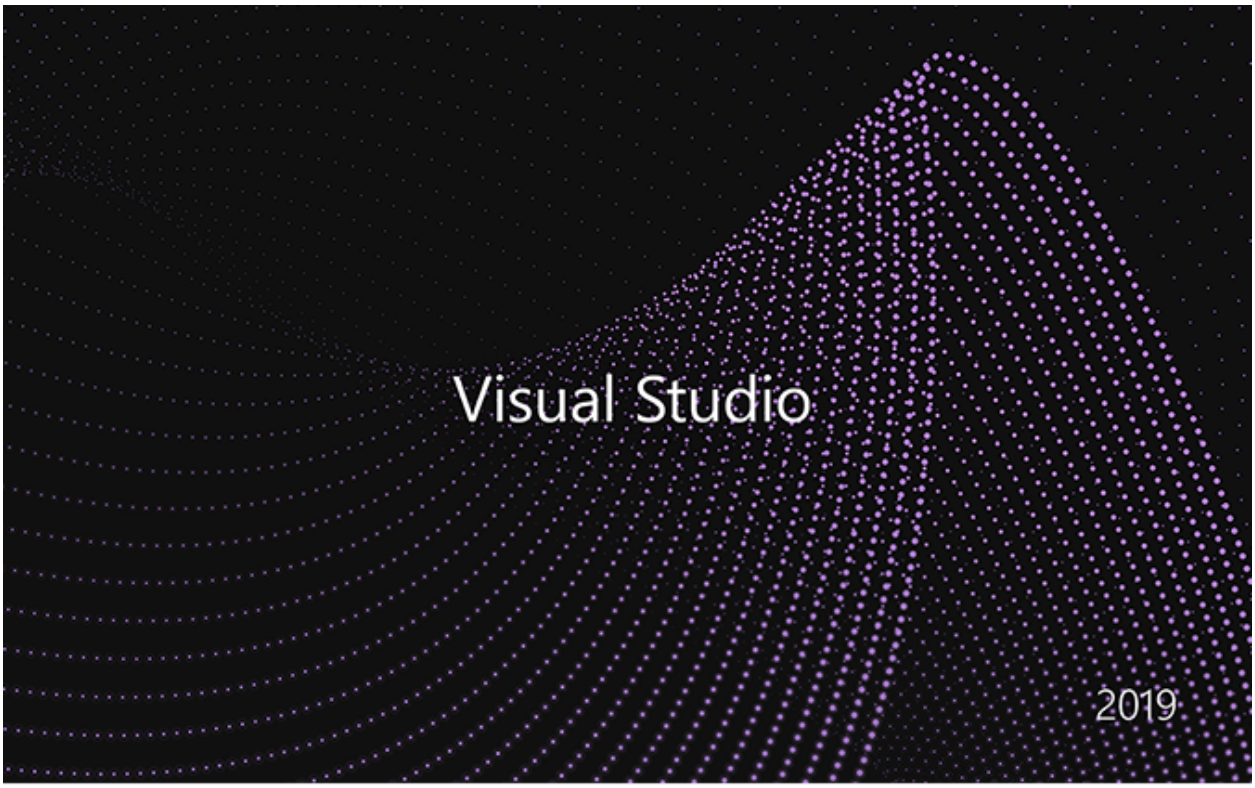
Visual Studio 2019 Community Edition - Son Ücretsiz Sürümü İndir[4]

Open visual studio community edition and select create a new project

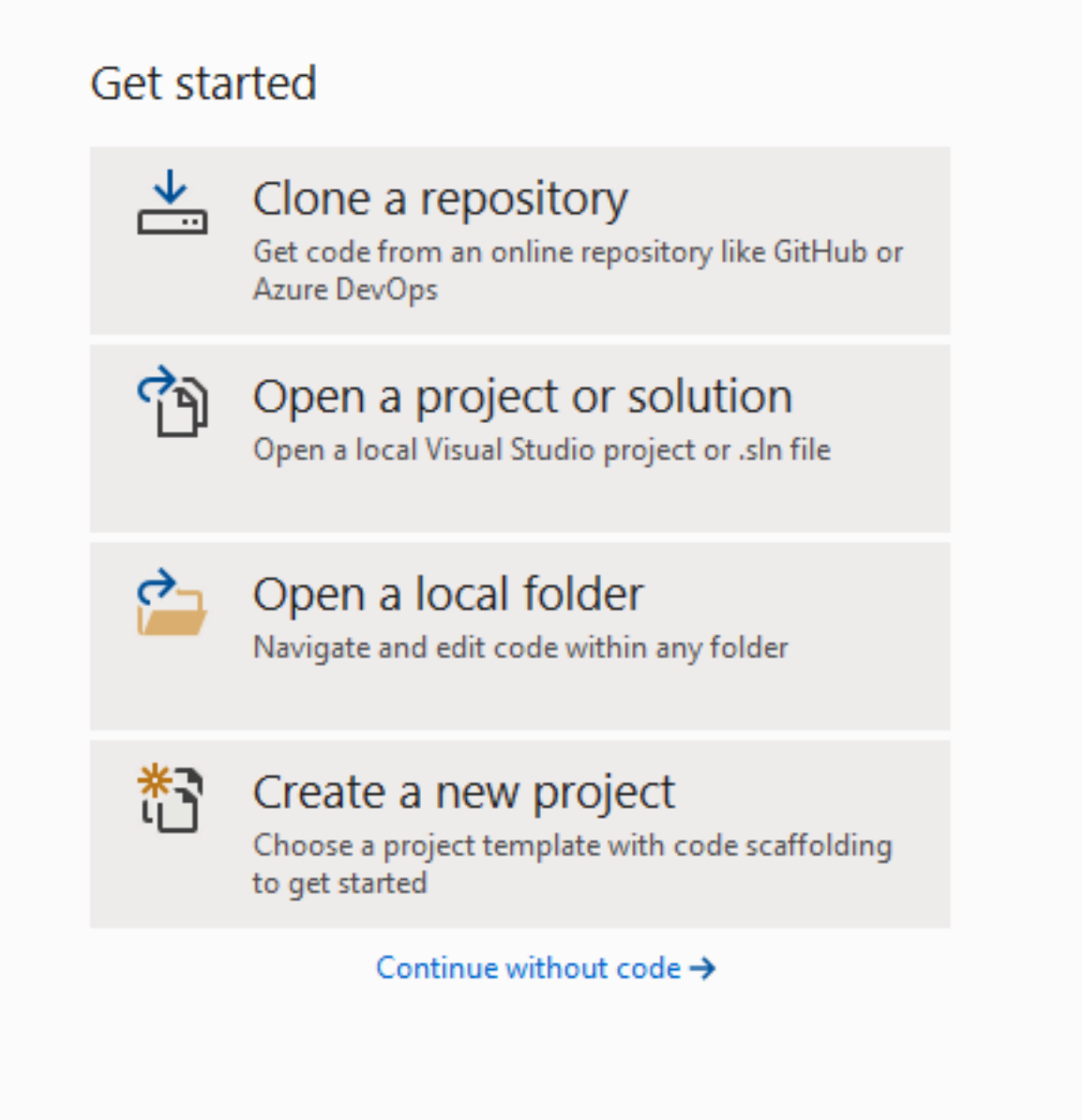


---

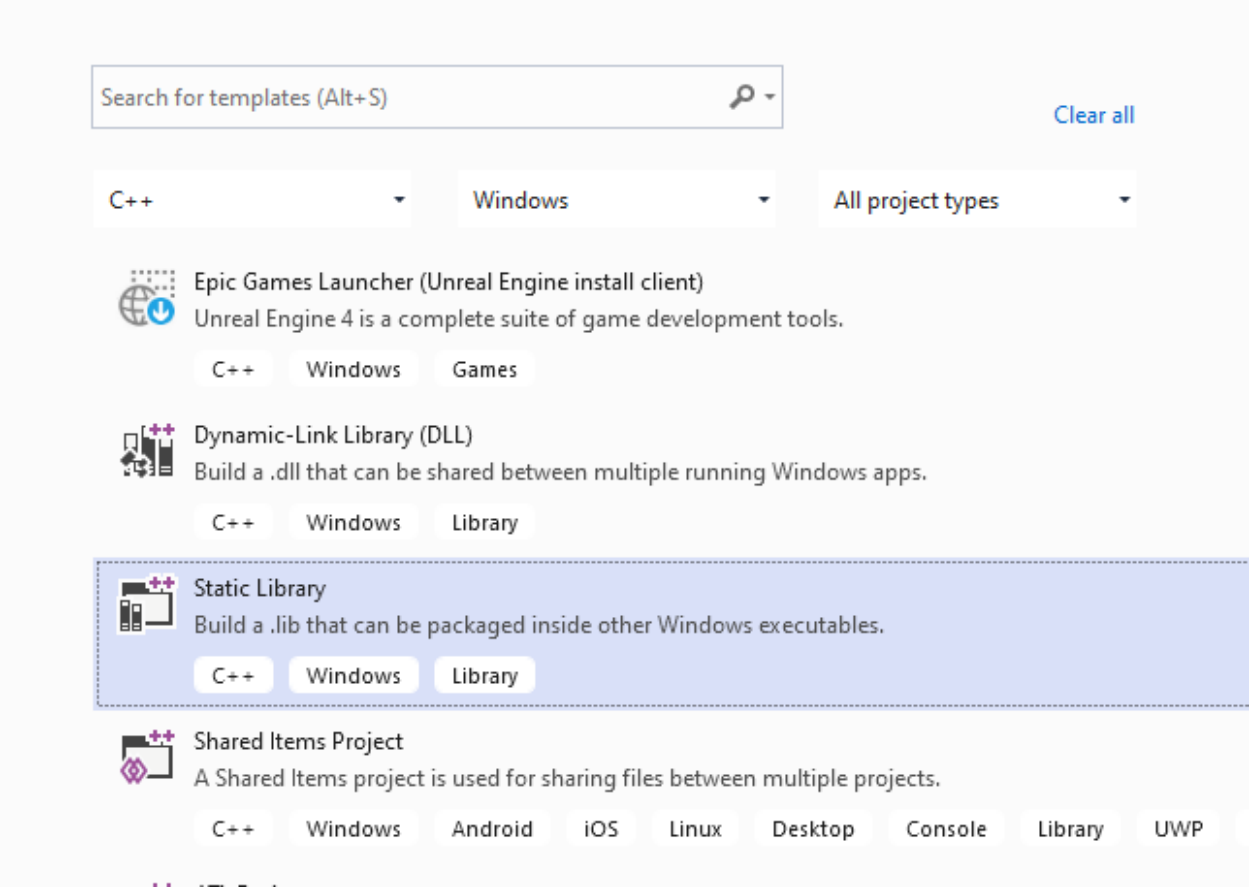
[4] https://visualstudio.microsoft.com/tr/vs/community/



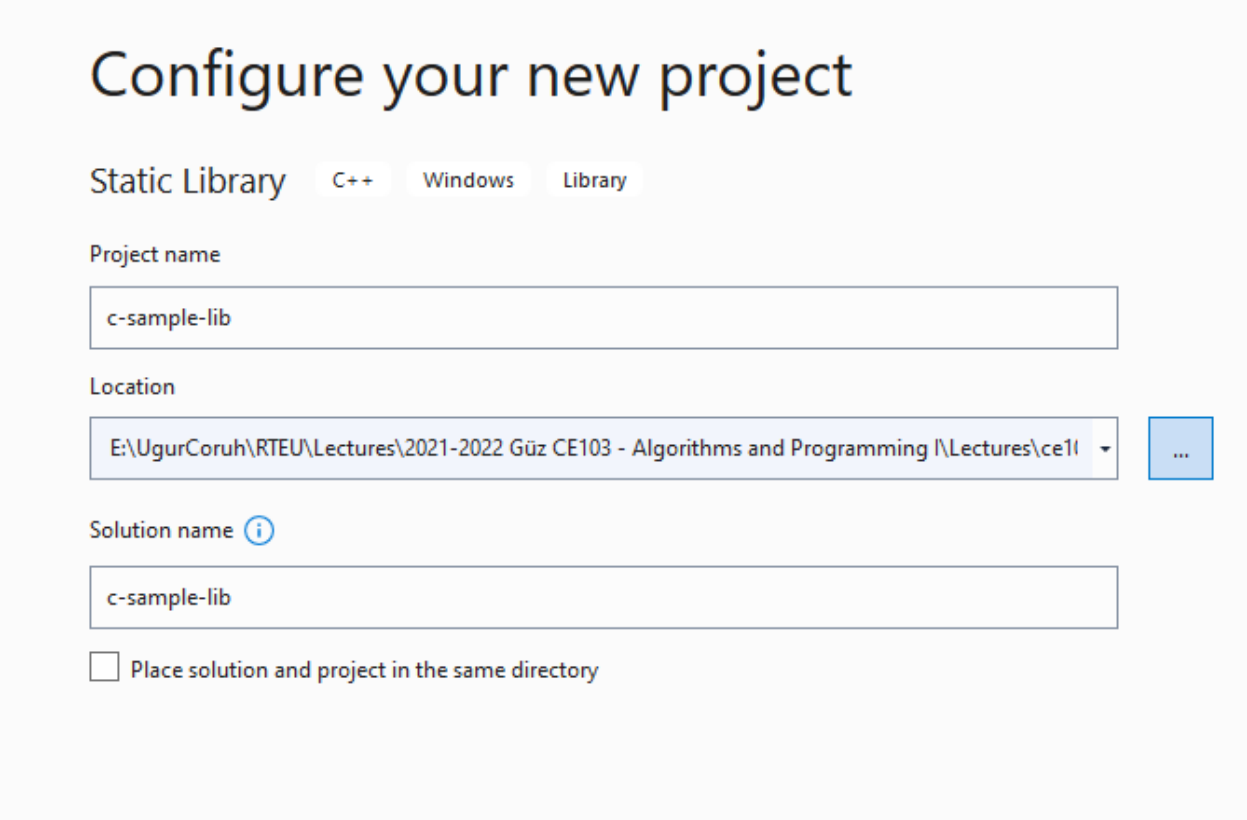
Select create a new project



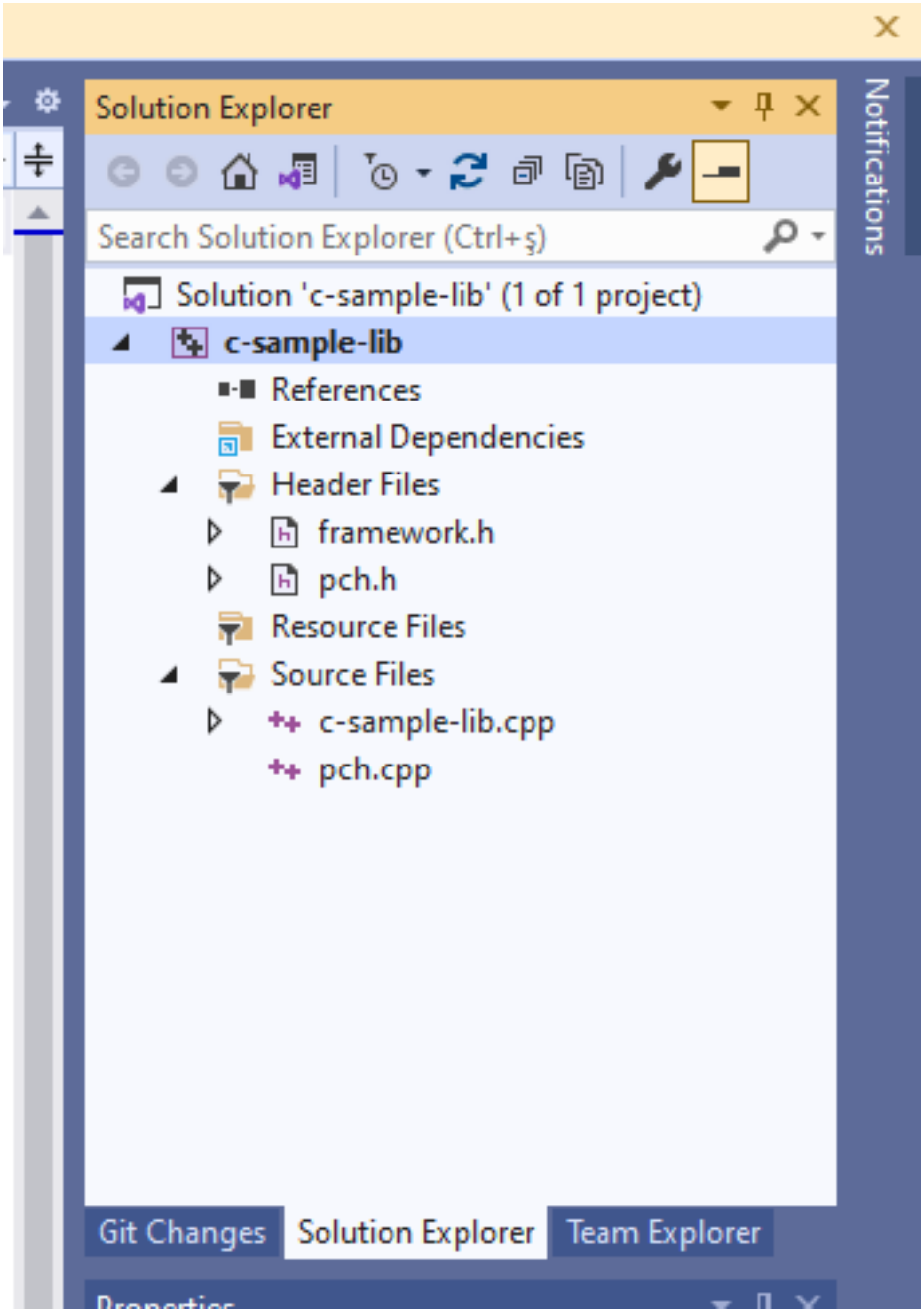
Select C++ static library from project list



Name static library project



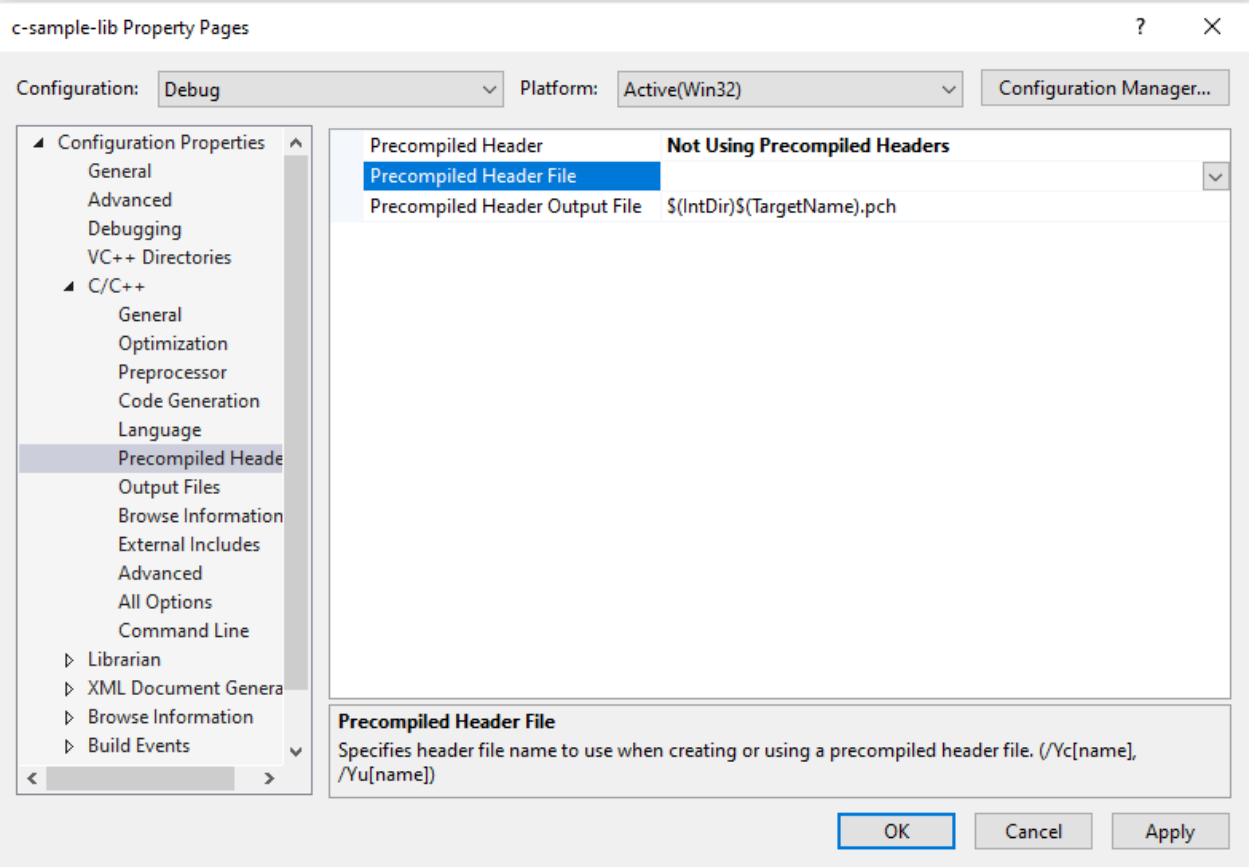
Default configuration come with C++ project types and setting



In the c-sample-lib.cpp you will sample function

```
void fncsamplelib()
{

}
```

Delete pch.h and pch.c files. Also disable use precompiled header settings from configurations and change to “Not Using Precomplied Headers”, also you can delete precomplied Header File.



Customize library header name and update “framework.h” to “samplelib.h”

Insert your functions inside the c-sample-lib.c and update header files also.

```
// c-sample-lib.cpp : Defines the functions for the static library.
//

#include "samplelib.h"
#include "stdio.h"

/// <summary>
///
/// </summary>
/// <param name="name"></param>
void sayHelloTo(char* name){

    if (name != NULL){
        printf("Hello %s \n",name);
    }
    else {
        printf("Hello There\n");
    }
}

/// <summary>
///
/// </summary>
/// <param name="a"></param>
```

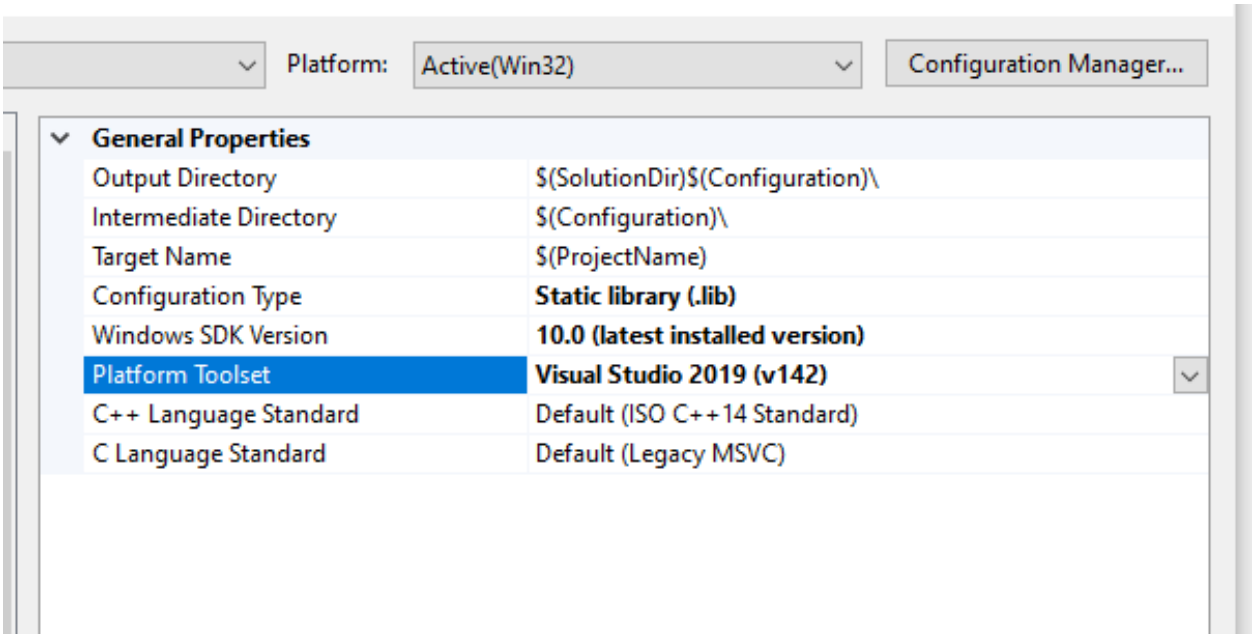
```
/// <param name="b"></param>
/// <returns></returns>
int sum(int a, int b){

    int c = 0;
    c = a + b;
    return c;
}
```

---

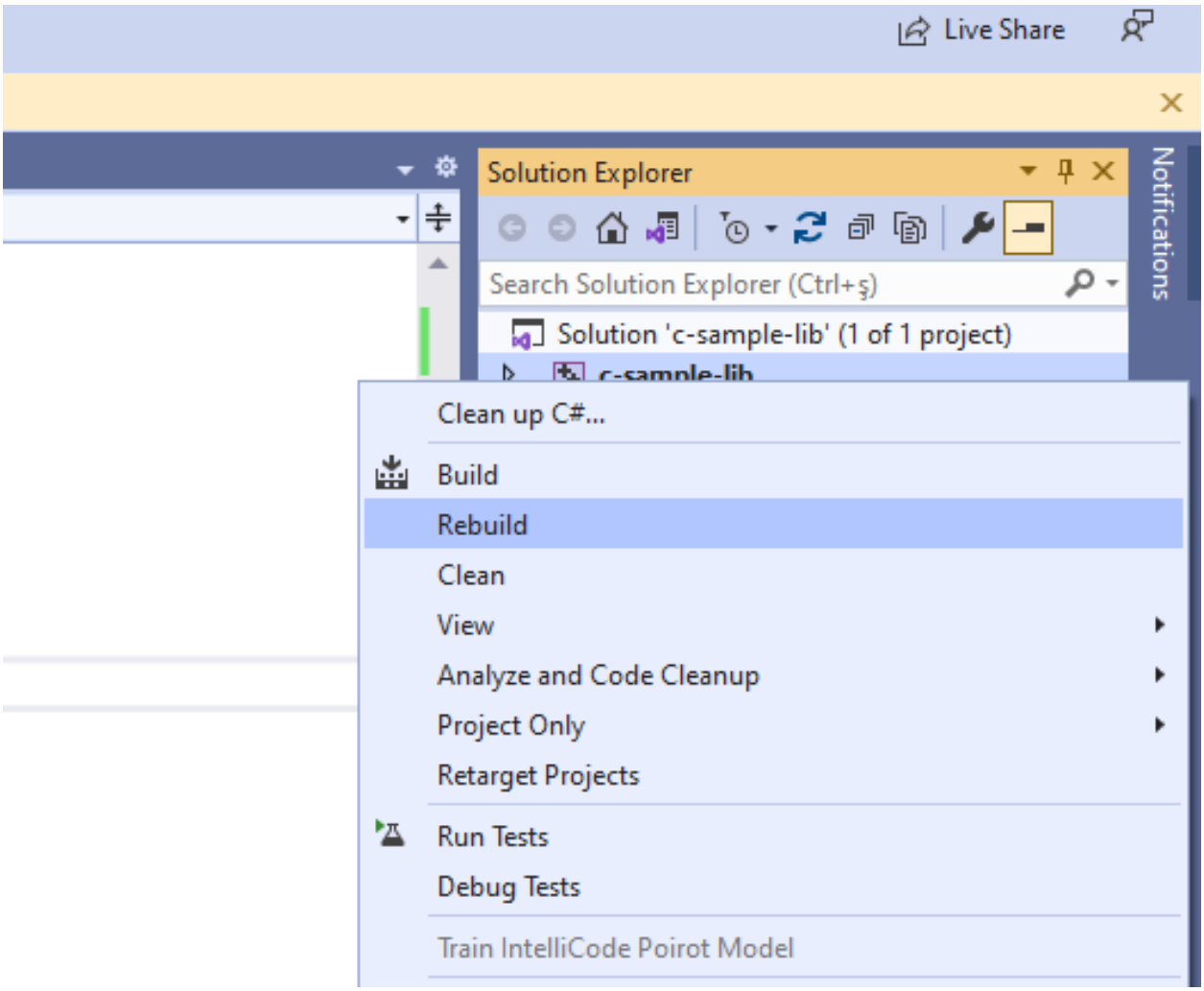
also update samplelib.h

```
#pragma once

#define WIN32_LEAN_AND_MEAN             // Exclude rarely-used stuff from Windows headers

void sayHelloTo(char* name);
int sum(int a, int b);
```

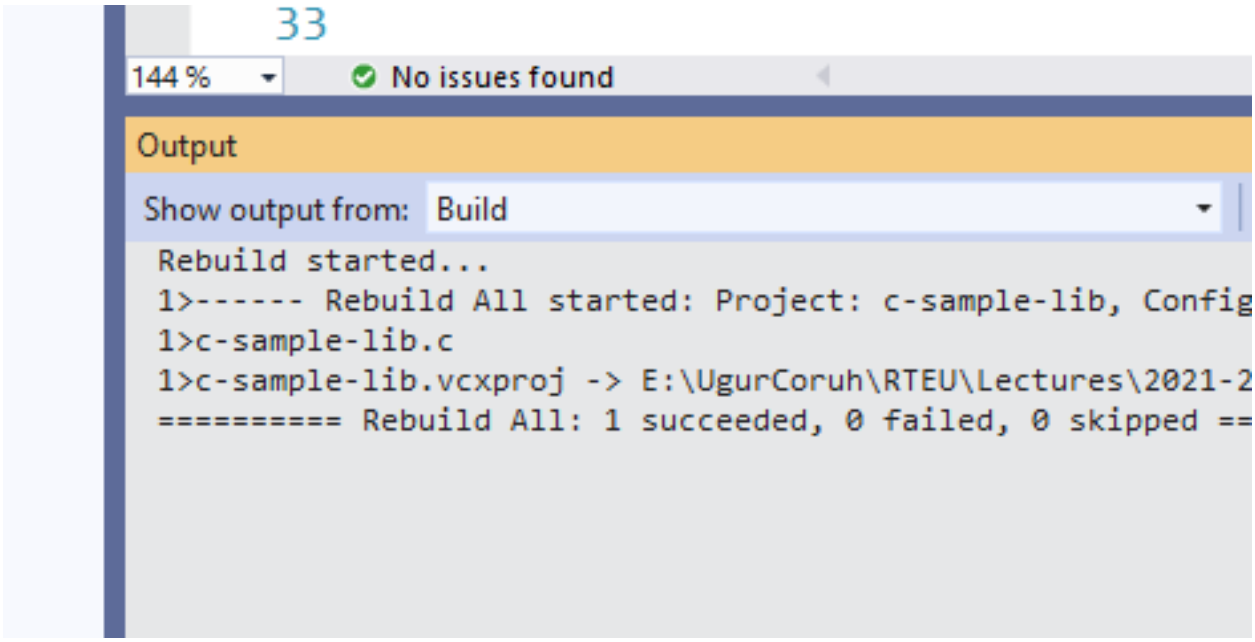
If you check configuration you will see that for C complier we are using Microsoft Environment and Toolkits



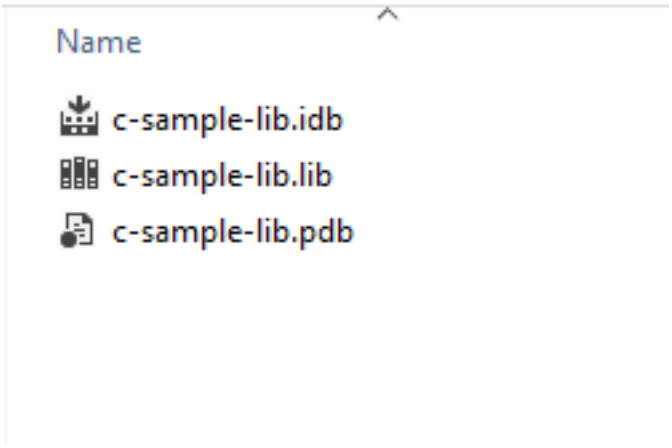
Now we can compile our library



You can follow operation from output window

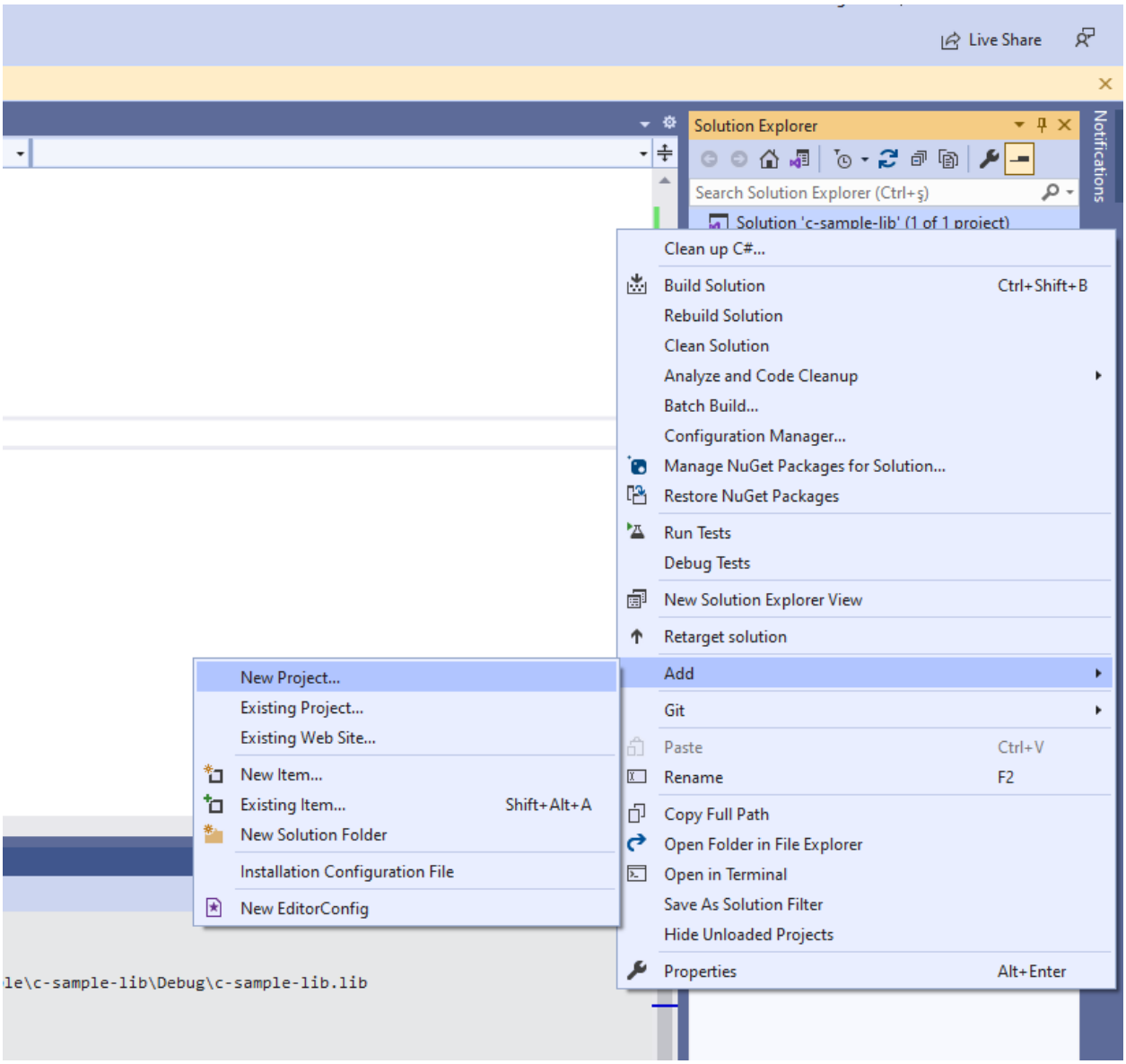


in debug folder we will see our output

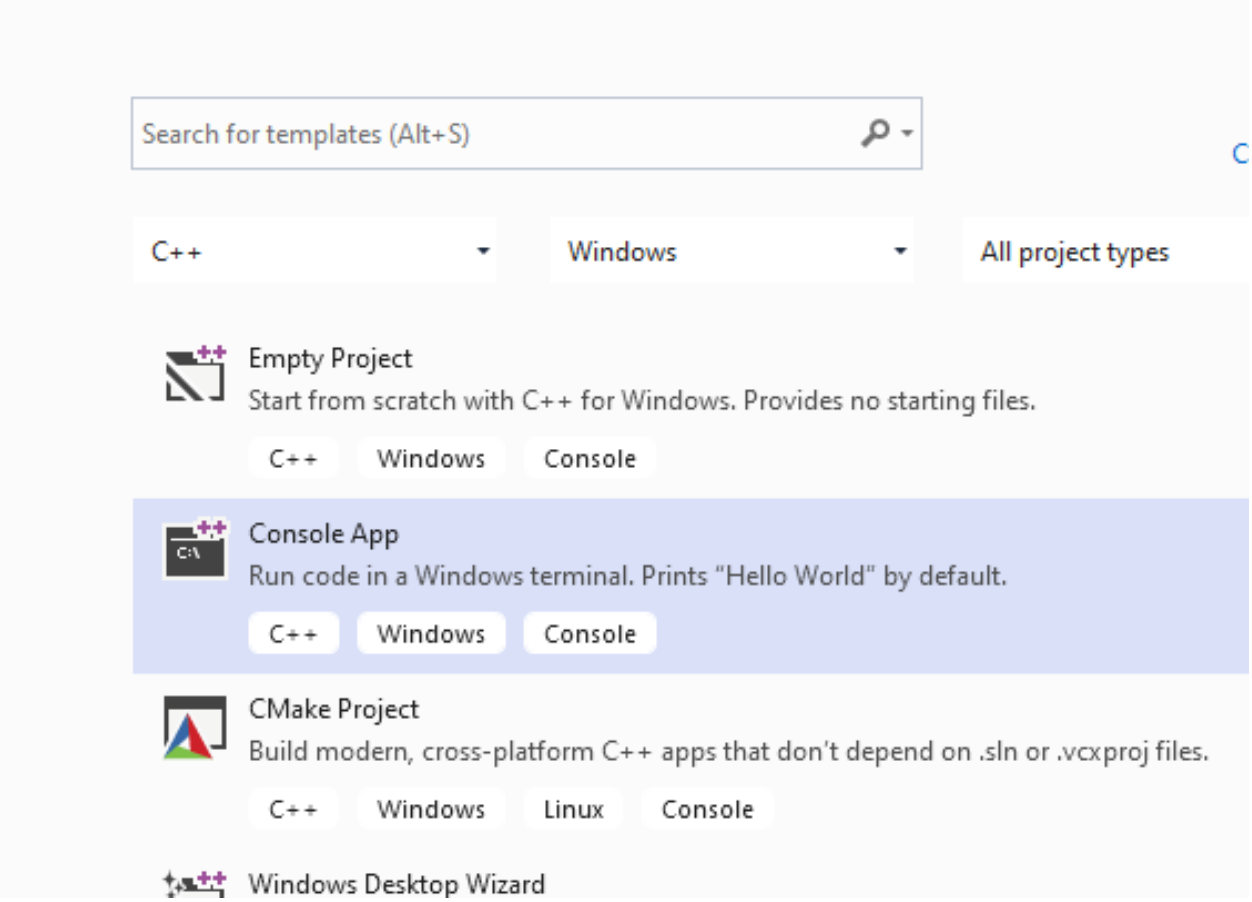


---

now we will add a console application c-sample-app and use our library



select C++ Windows Console Application from list



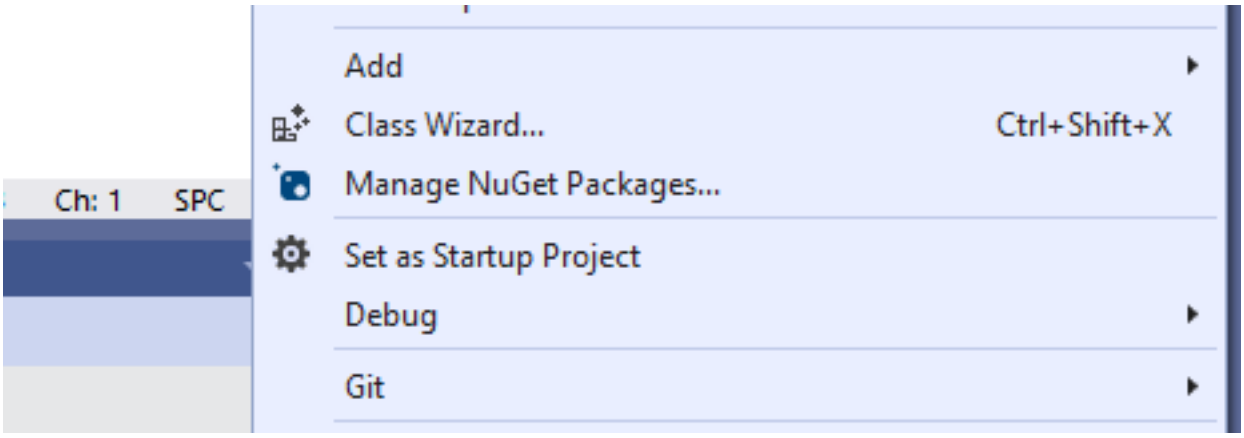
C++ Console Application Selection will generate a C++ console project we can change extension to C to compile our application as C application.

we will convert c-sample-app.c to following code

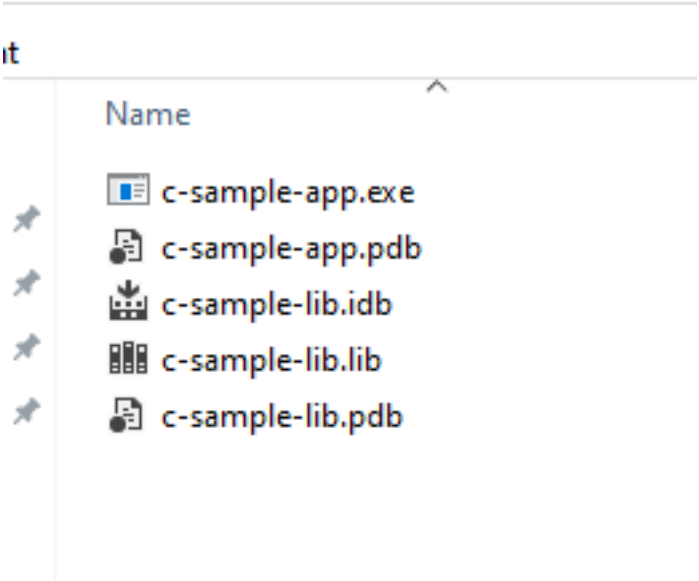
```c
#include <stdio.h>

/// <summary>
///
/// </summary>
/// <returns></returns>
int main()
{
    printf("Hello World!\n");
}
```

---

after conversion set c-sample-app as startup project and build it



this will create c-sample-app.exe in the same folder with c-sample-lib.lib library



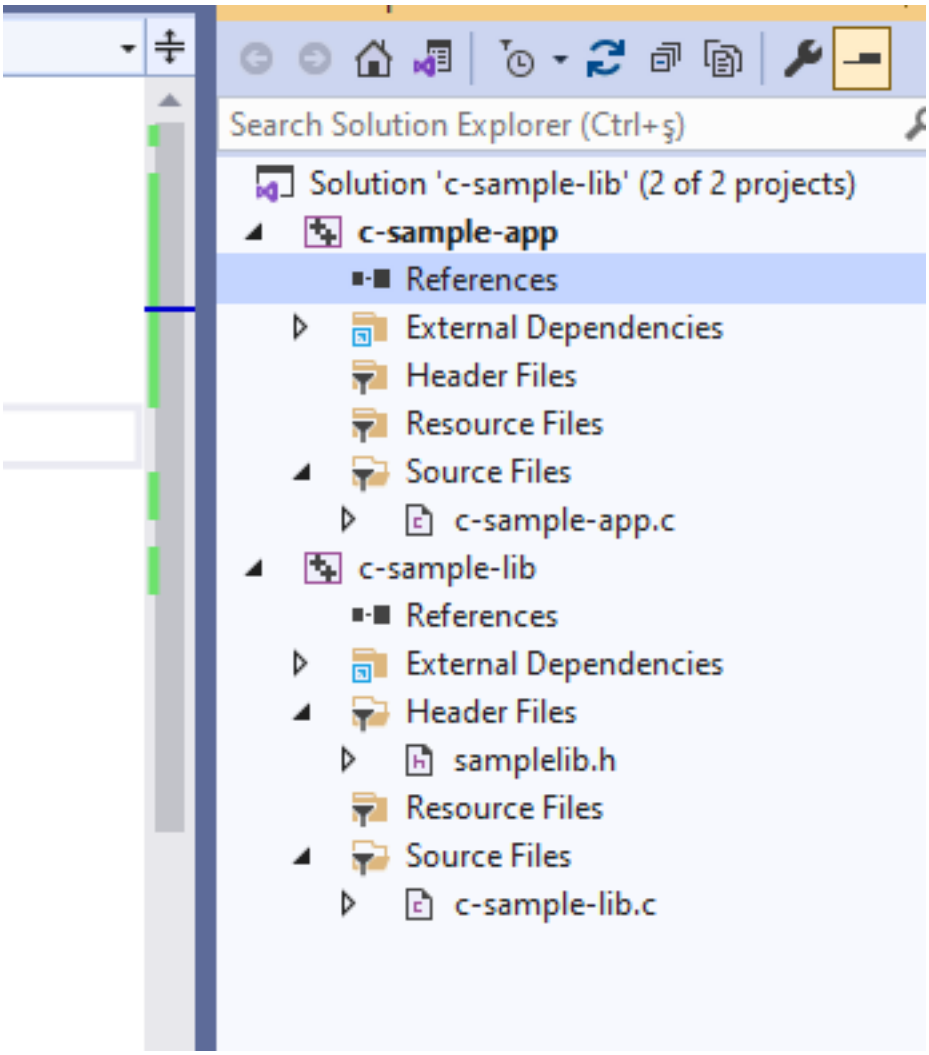
if we run application we will see only "Hello World"

---

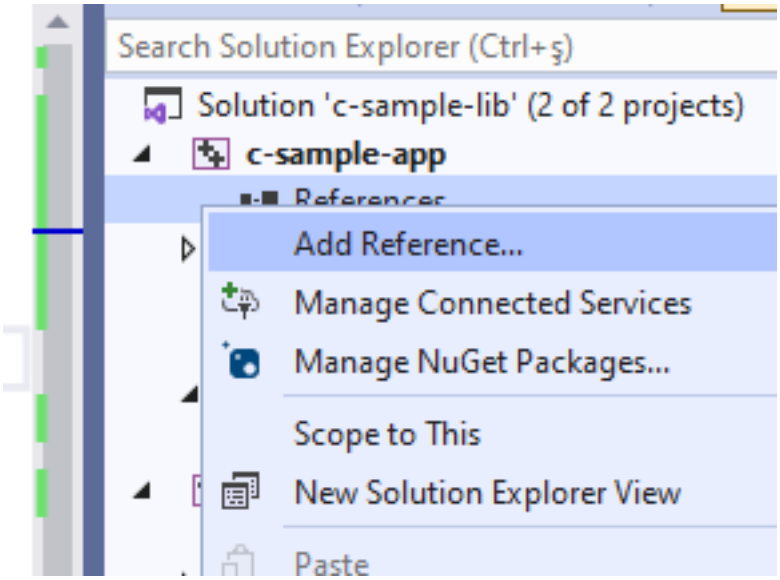
now we will see two options to add library as references in our application and use its functions.

First option

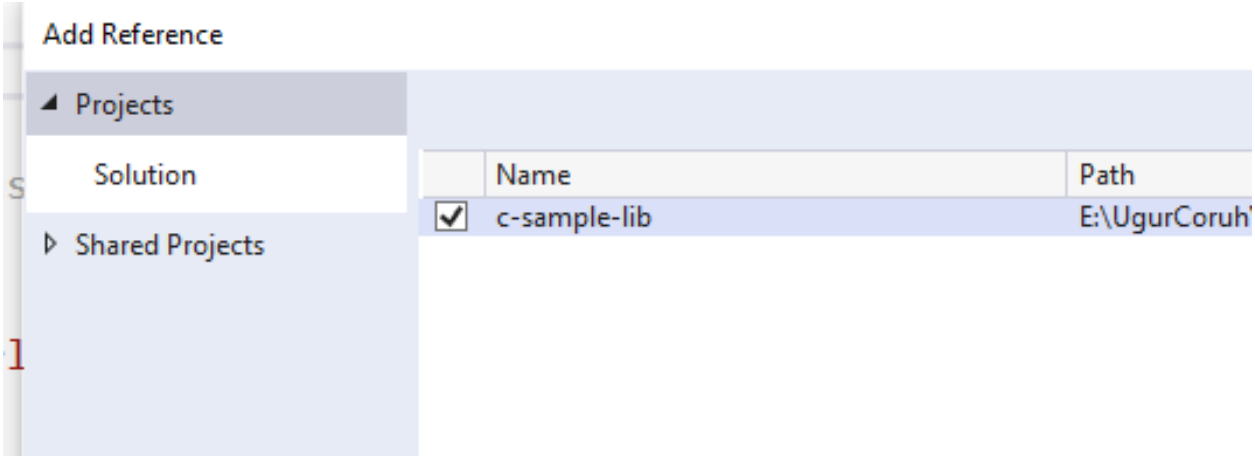
right click references for c-sample-app and add current library as reference



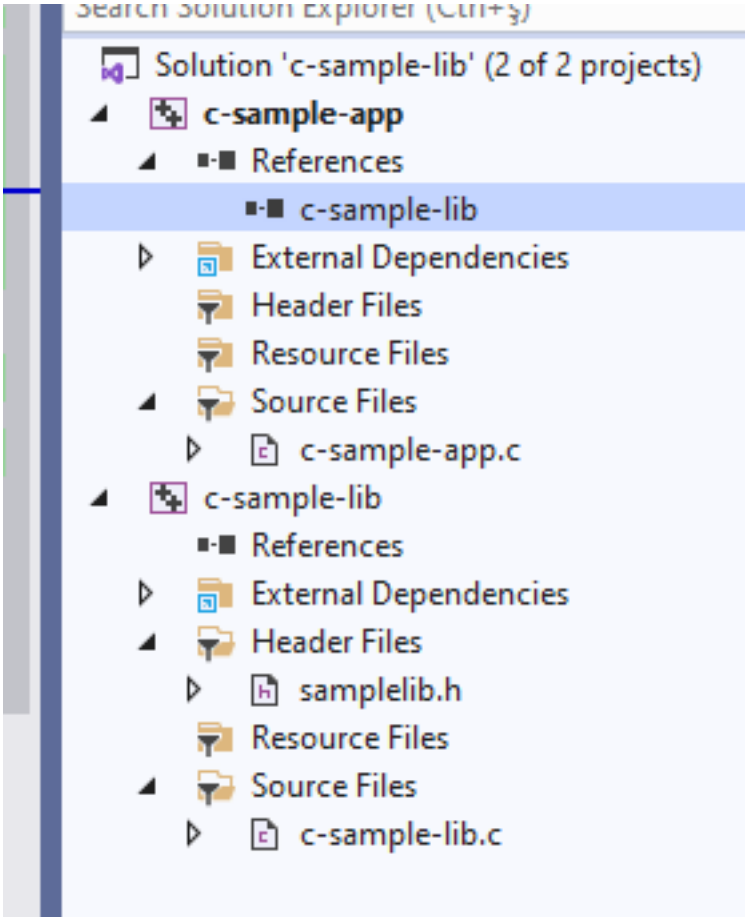
Select Add Reference



Browse for solution and select c-sample-lib



You can check added reference from references section



---

now we can include required headers from c-sample-lib folder and use it.

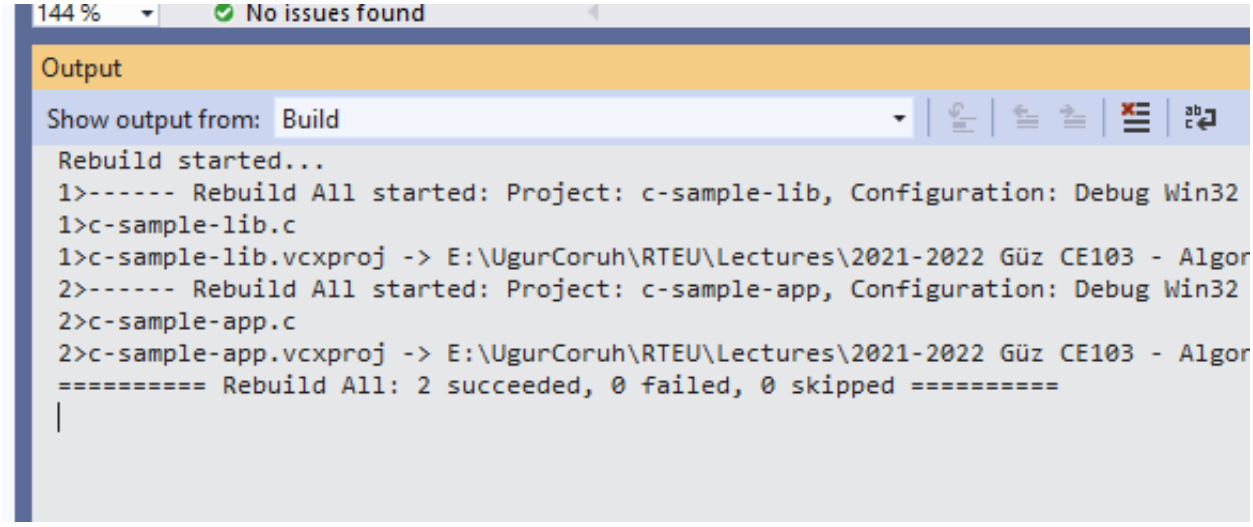
we can include required header with relative path as follow or with configuration

```
#include <stdio.h>
#include "..\c-sample-lib\samplelib.h"
/// <summary>
///
```

```
/// </summary>
/// <returns></returns>
int main()
{
    printf("Hello World!\n");
}
```

---

we can build our c-sample-app



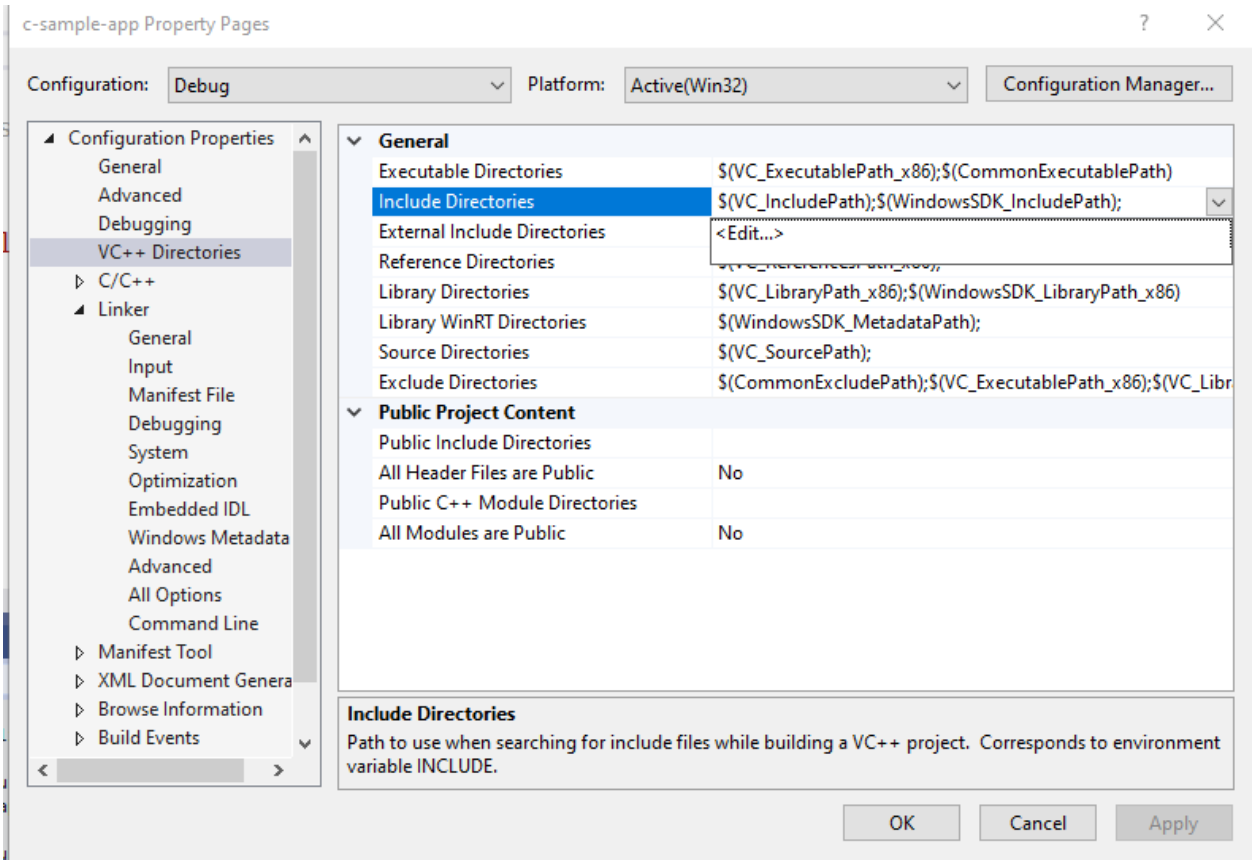
---

also we can only write header name

```
#include <samplelib.h>
```

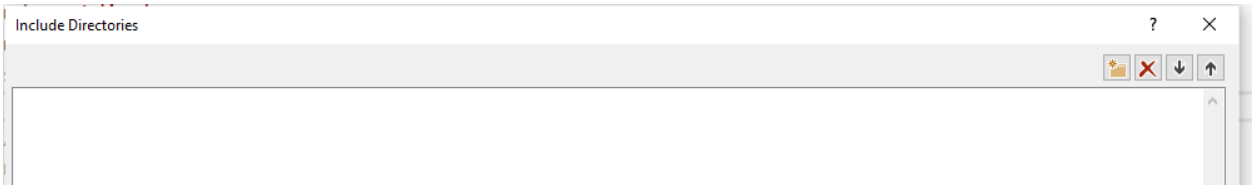
---

for this we need to configure include directories

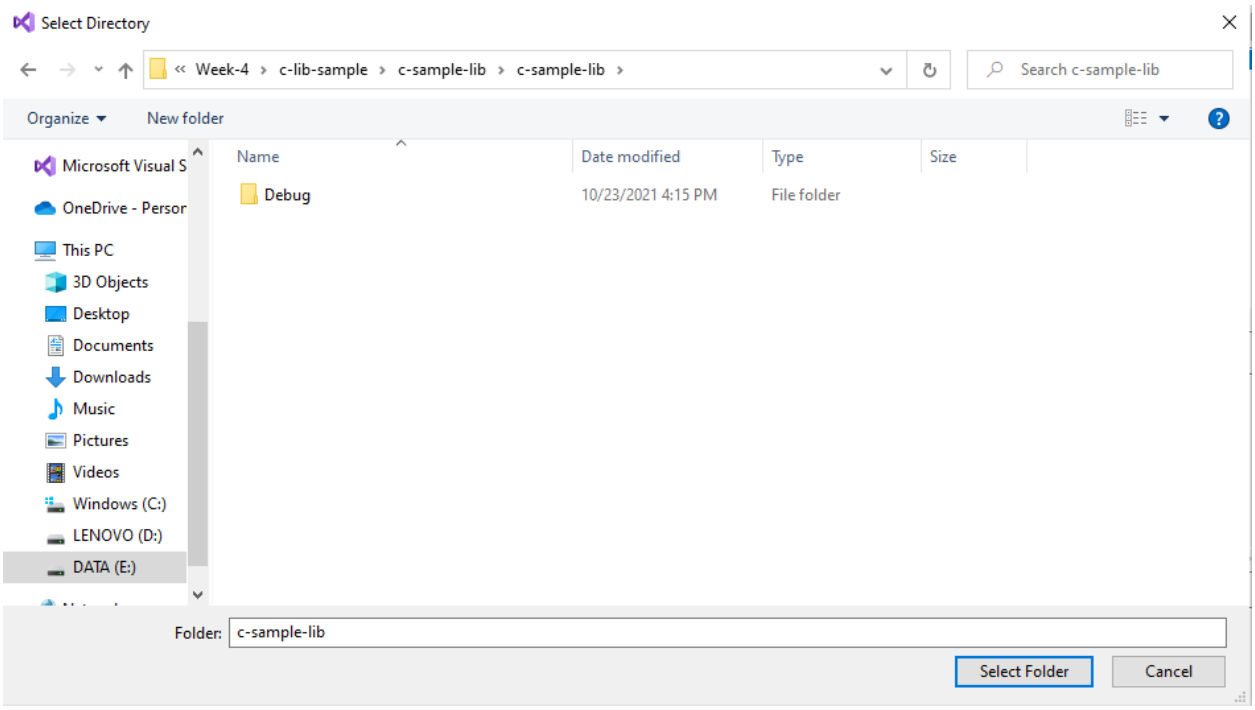


---

select c-sample-lib header file location

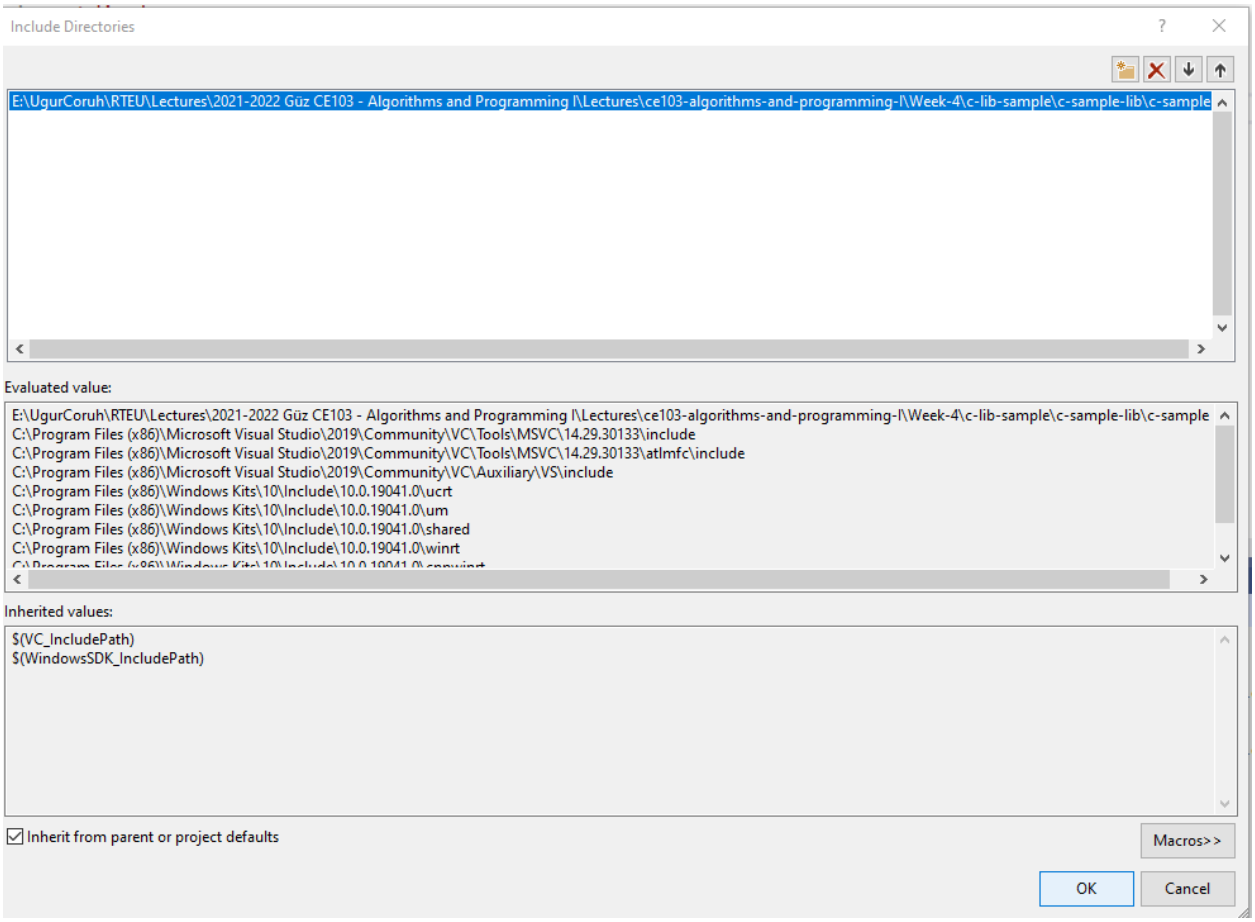


browse for folder



---

your full path will be added to your configuration



---

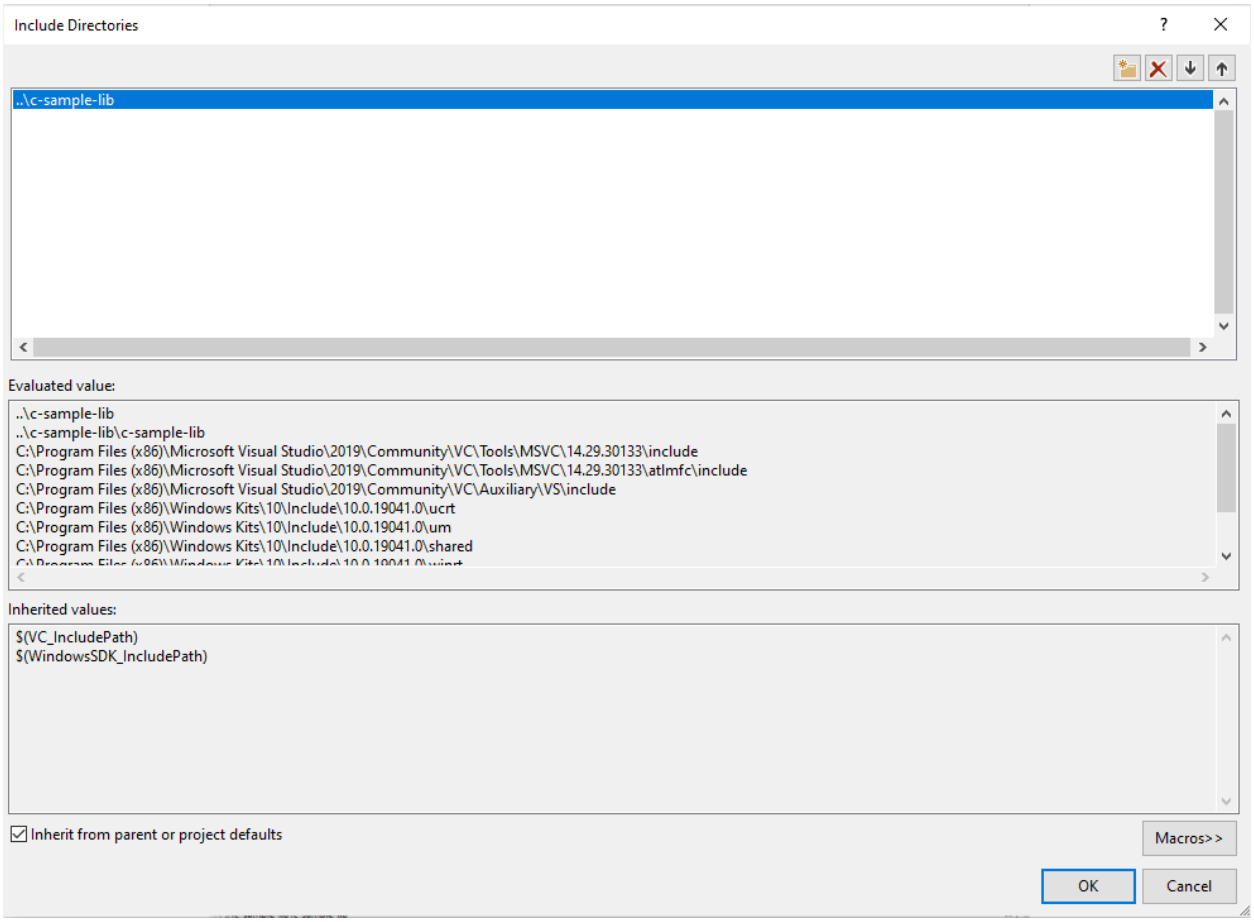
if you add header file paths to your configuration you can use header files by name in your source code

```
#include <stdio.h>
#include <samplelib.h>
/// <summary>
///
/// </summary>
/// <returns></returns>
int main()
{
    printf("Hello World!\n");
}
```

---

we can compile the following we don't have problems but here we need to configure relative paths for configuration open include library settings and update with relative path

`..\c-sample-lib`



---

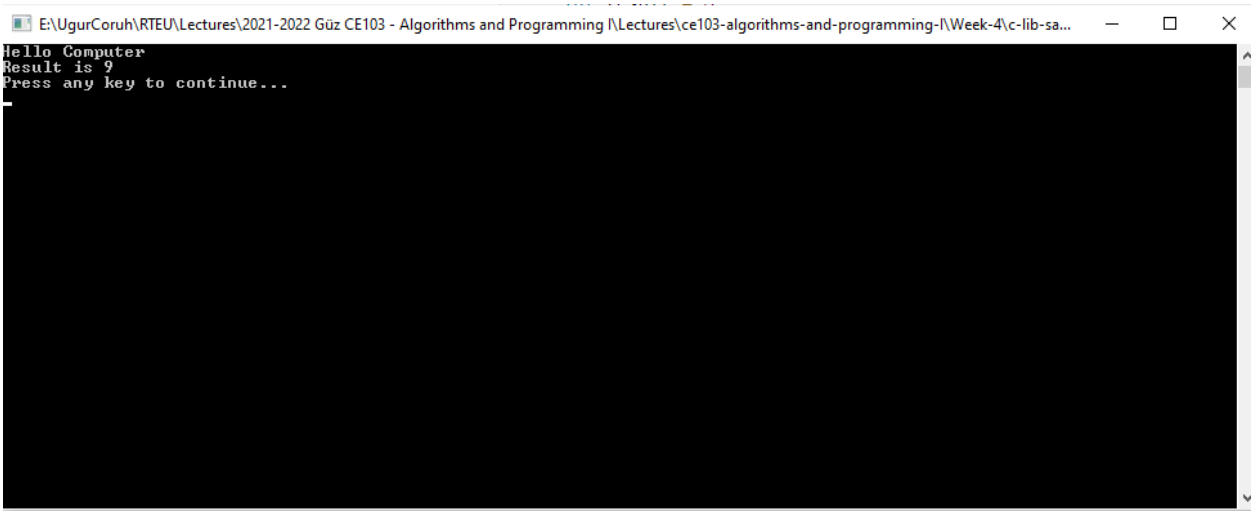
now we have portable source code configuration. we can call our functions and then we can update header and library folder configurations.

```
#include <stdio.h>
#include <samplelib.h>
/// <summary>
///
/// </summary>
/// <returns></returns>
```

```c
int main()
{
    int result = 0;
    //printf("Hello World!\n");
    result = sum(5, 4);
    sayHelloTo("Computer");
    printf("Result is %d \n",result);
    printf("Press any key to continue...\n");
    getchar();
    return 0;
}
```

---

when you run you will see the following outputs, that mean we called library functions.



---

static library is a code sharing approach if you want to share your source code with your customers then you can share static libraries and header files together. Another case you can use a precompiled static library with you or this library can be part of any installation then if there is a installed app and static libraries are placed on system folder or any different location then you can use configuration files to set library path and included header paths

---

Now we can remove project from c-sample-app references but we will set library file in configuration

Before this copy static library and header files to a folder like that

`DebugStaticLibDeployment`

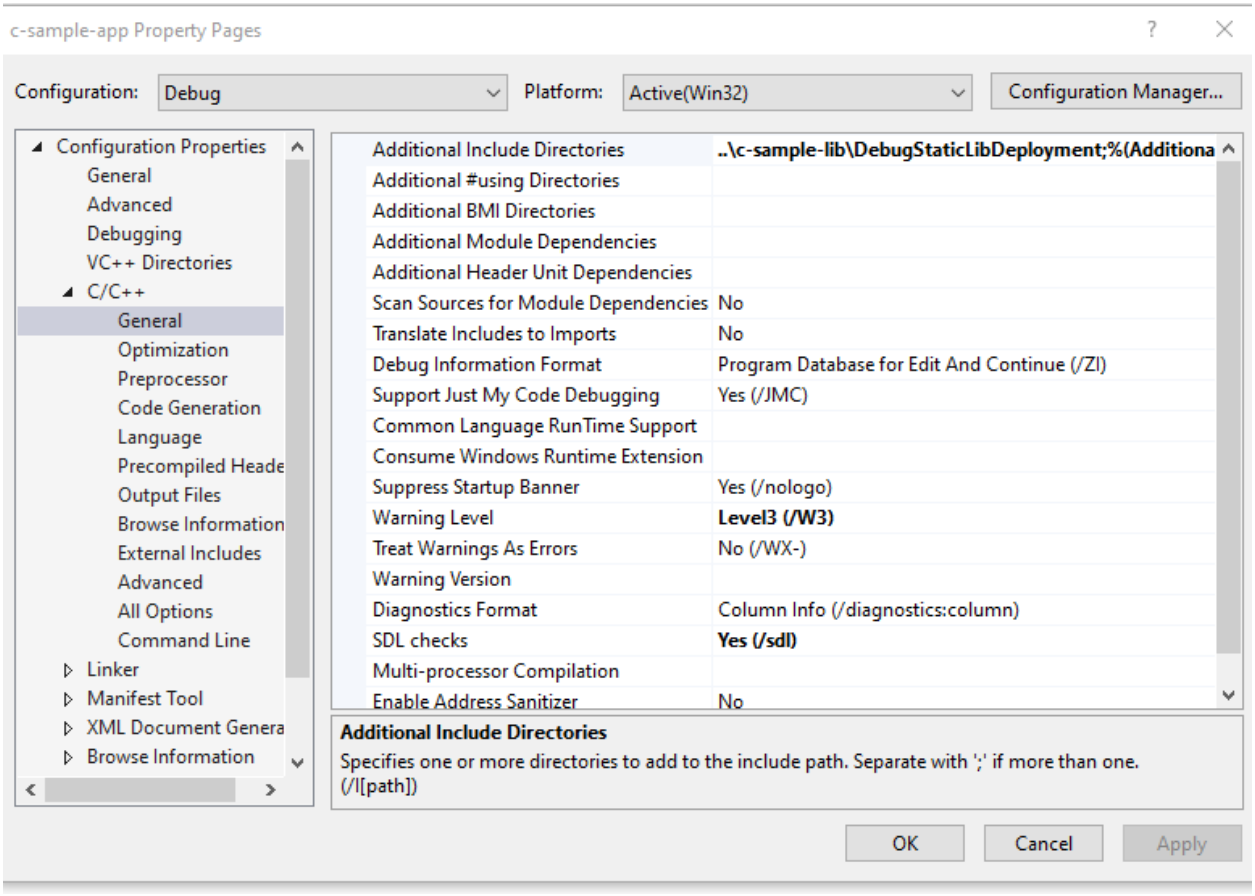
- Set C/C++ -> General -> Additional Include Directories

There is a bug in configurations and relative path not finding headers so for this reason we will set full path but this is not a good practice for team working

---

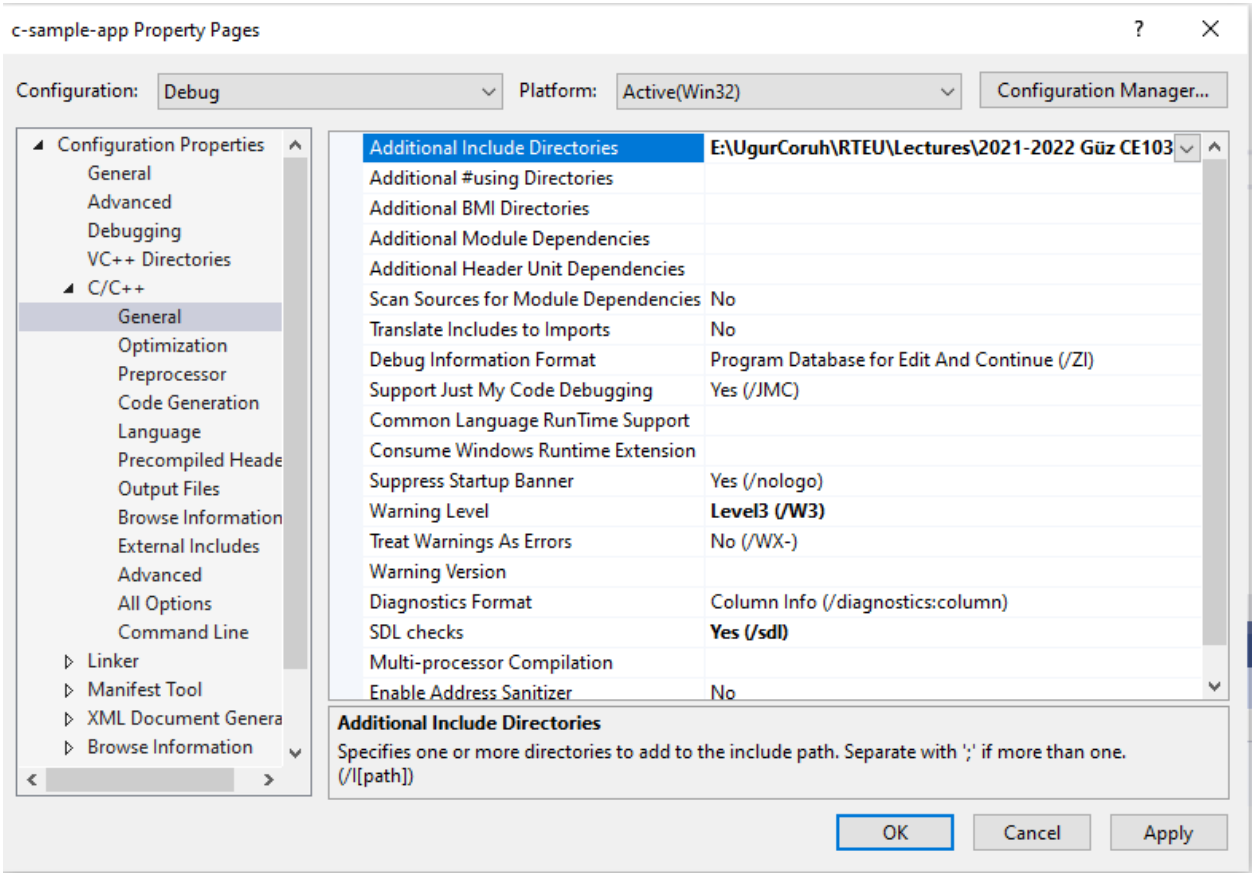
Not Working

`..\c-sample-lib\DebugStaticLibDeployment`



Working

`E:\...\c-lib-sample\c-sample-lib\DebugStaticLibDeployment`



Now we will set library folder that our static library placed

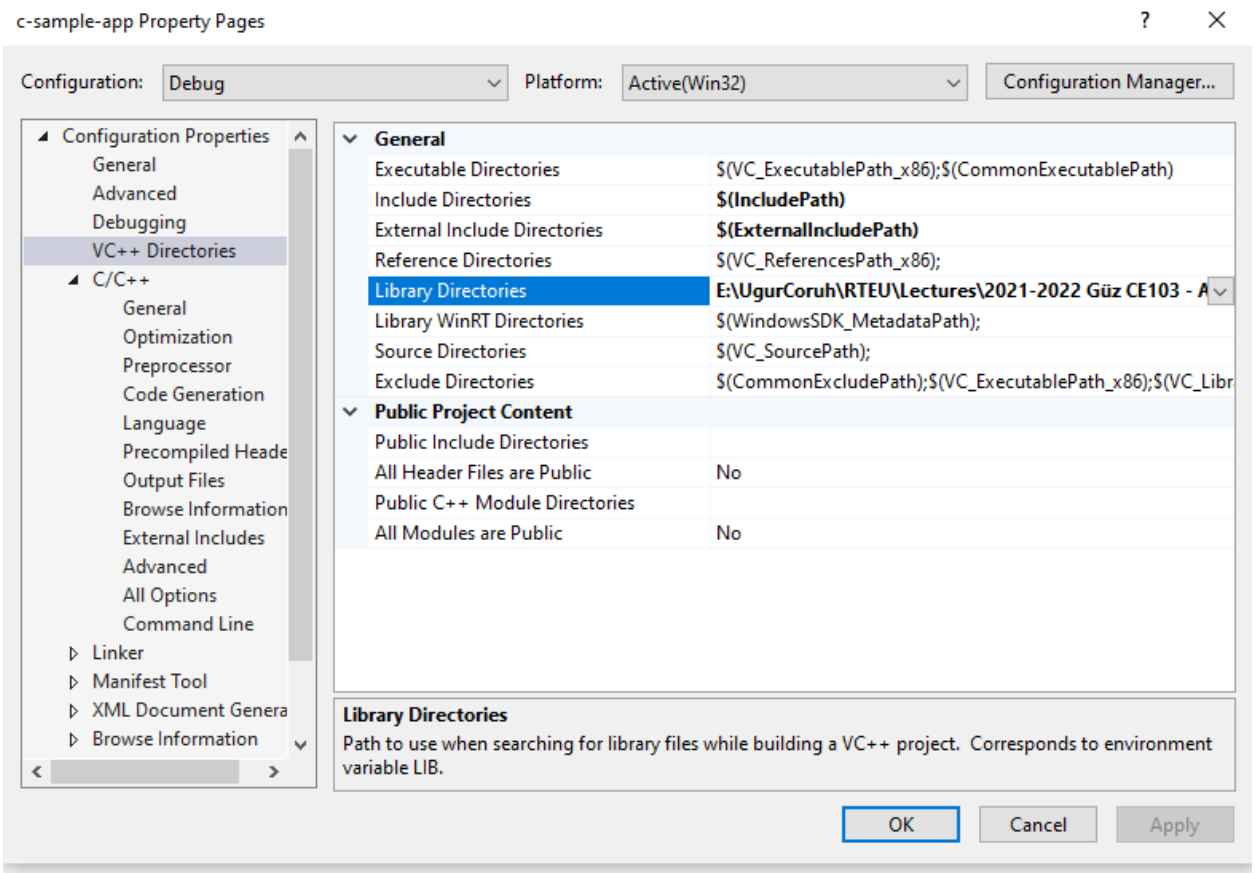
we will set VC++ Directories -> Library Directories

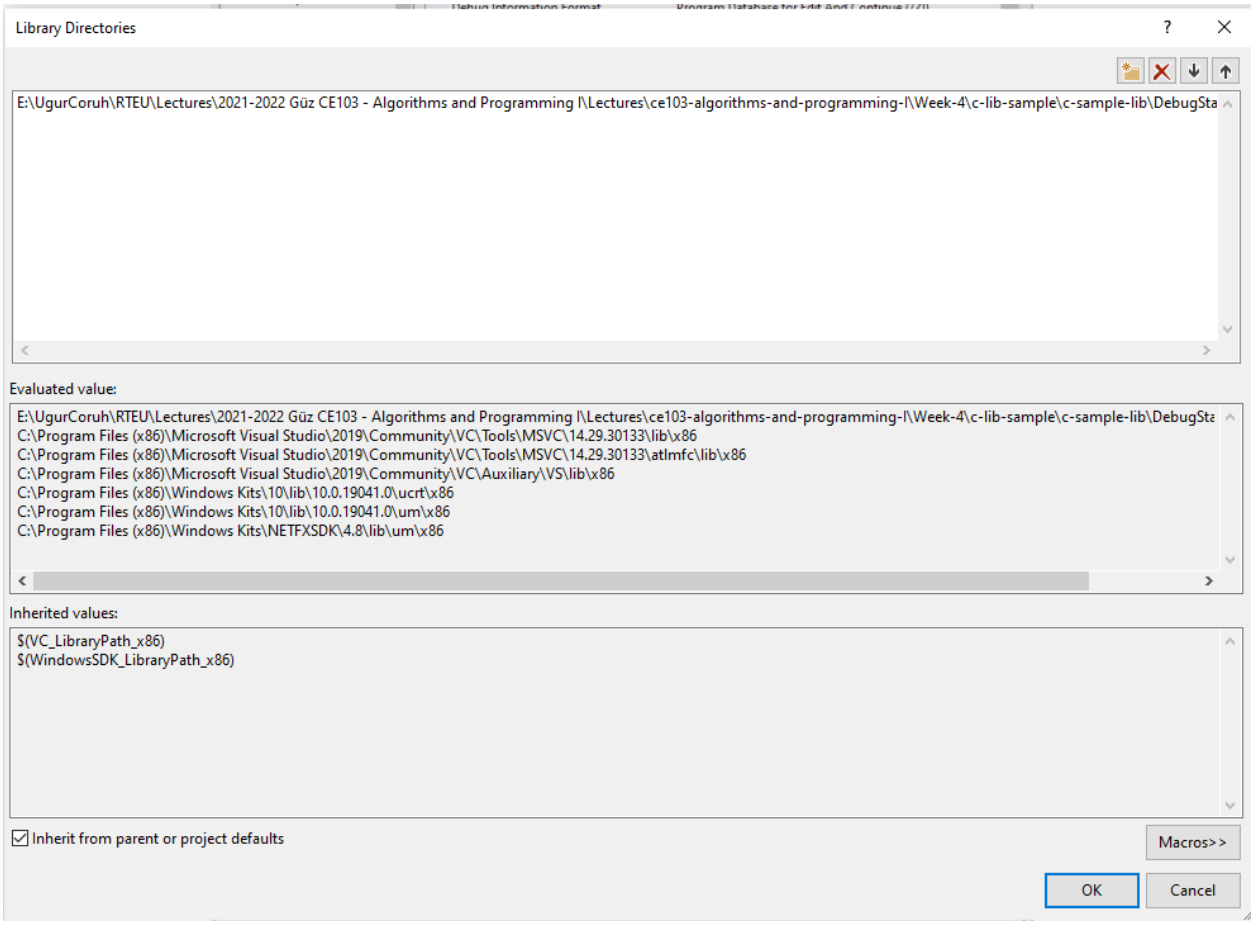
Here is the same issue if we use relative path it doesn't work we need to set full path for library folder

---

Working

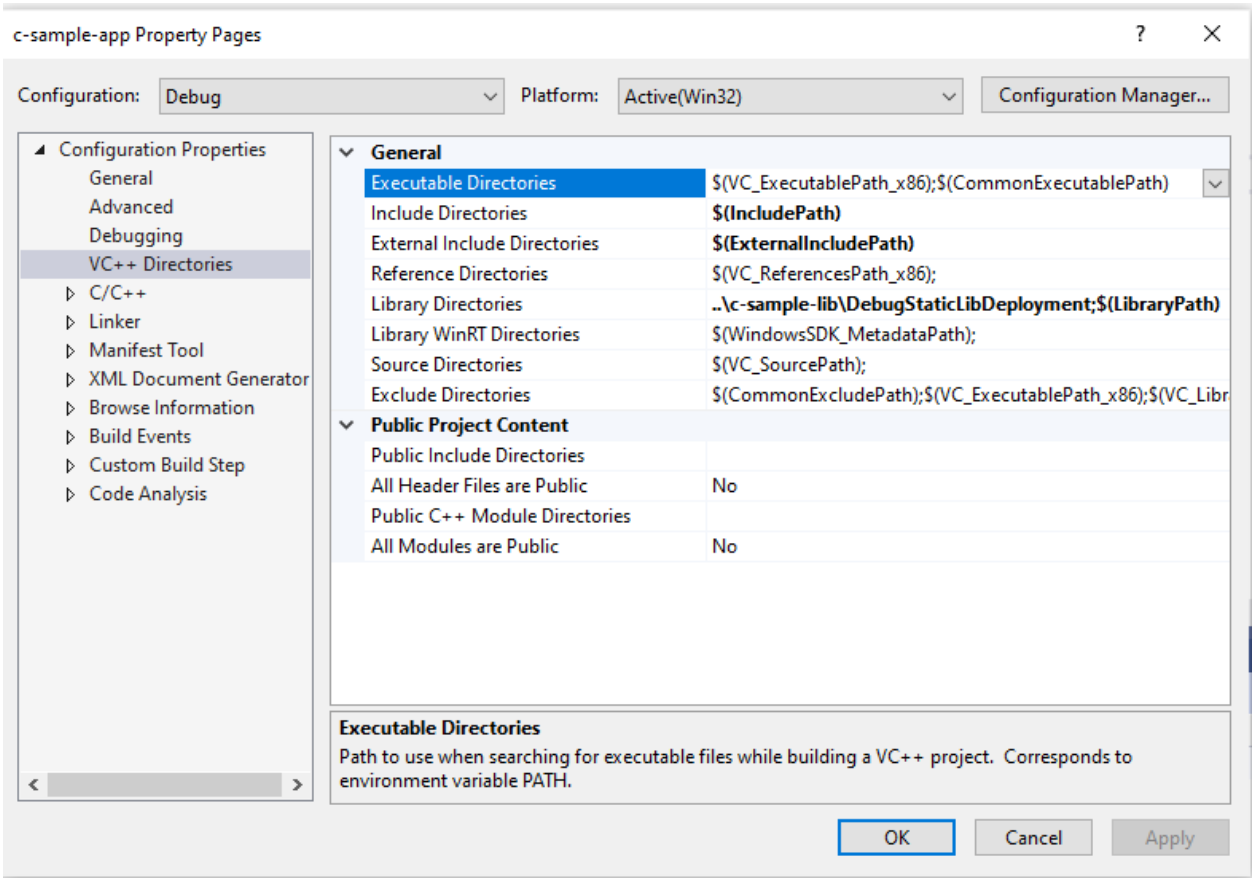
`E:\...\c-lib-sample\c-sample-lib\DebugStaticLibDeployment`

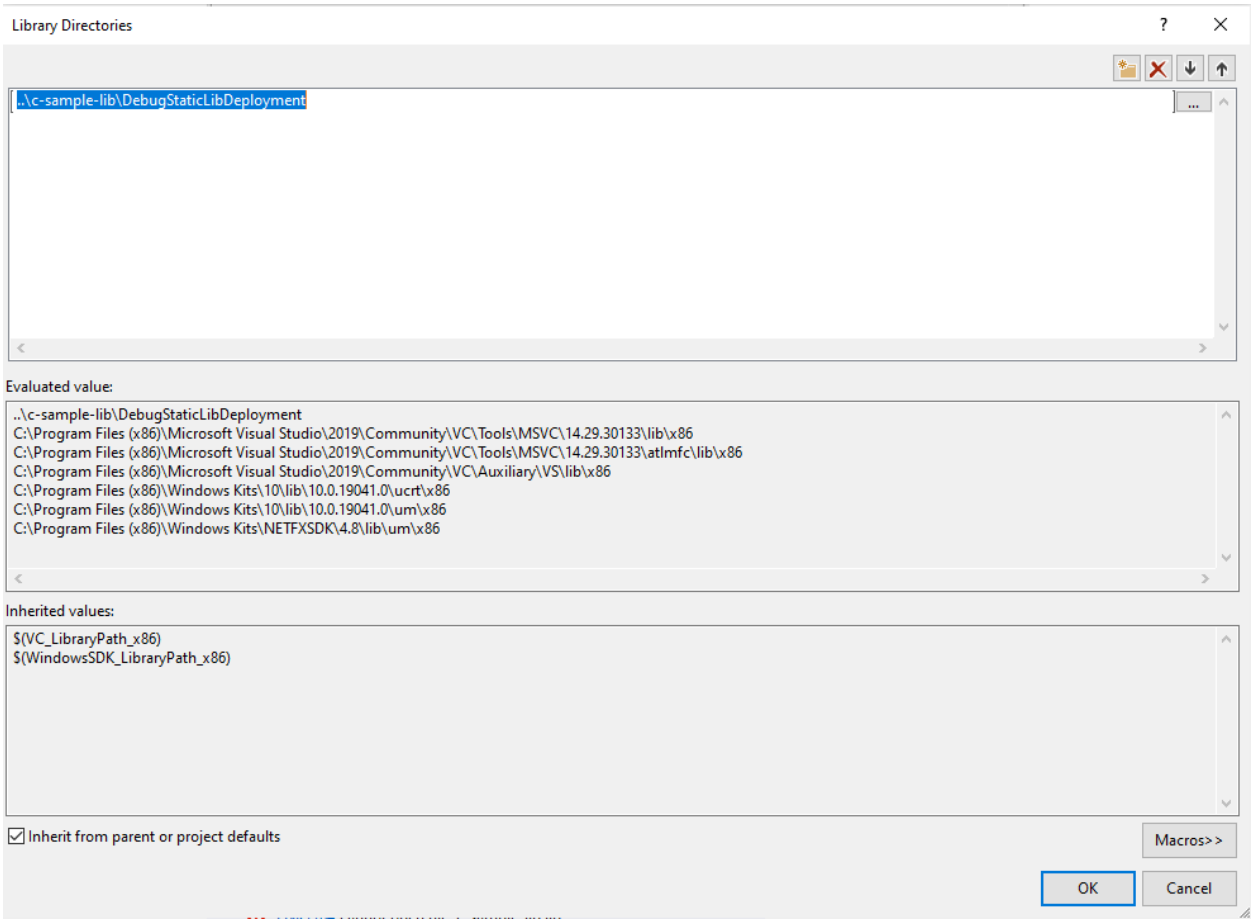

c-sample-app Property Pages
Configuration: Debug
Platform: Active(Win32)
Configuration Manager...
Configuration Properties
General
Advanced
Debugging
VC++ Directories
C/C++
General
Optimization
Preprocessor
Code Generation
Language
Precompiled Heade
Output Files
Browse Information
External Includes
Advanced
All Options
Command Line
Linker
Manifest Tool
XML Document Genera
Browse Information
General
Executable Directories $(VC_ExecutablePath_x86);$(CommonExecutablePath)
Include Directories $(IncludePath)
External Include Directories $(ExternalIncludePath)
Reference Directories $(VC_ReferencesPath_x86);
Library Directories E:\UgurCoruh\RTEU\Lectures\2021-2022 Güz CE103 - A
Library WinRT Directories $(WindowsSDK_MetadataPath);
Source Directories $(VC_SourcePath);
Exclude Directories $(CommonExcludePath);$(VC_ExecutablePath_x86);$(VC_Libr
Public Project Content
Public Include Directories
All Header Files are Public No
Public C++ Module Directories
All Modules are Public No
Library Directories
Path to use when searching for library files while building a VC++ project. Corresponds to environment variable LIB.
OK
Cancel
Apply

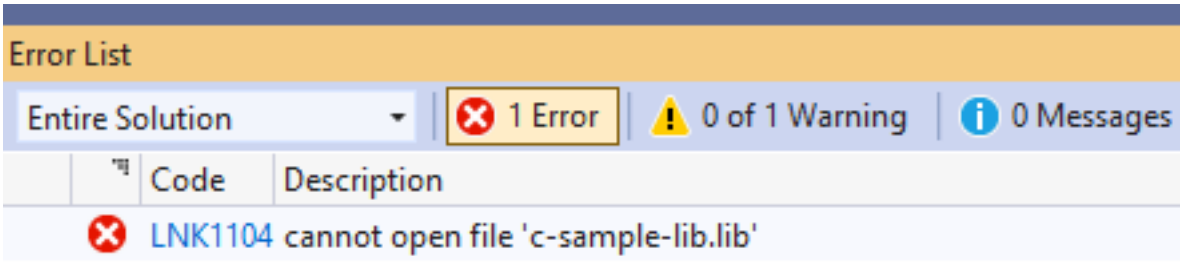


Not Working

`..\c-sample-lib\DebugStaticLibDeployment`

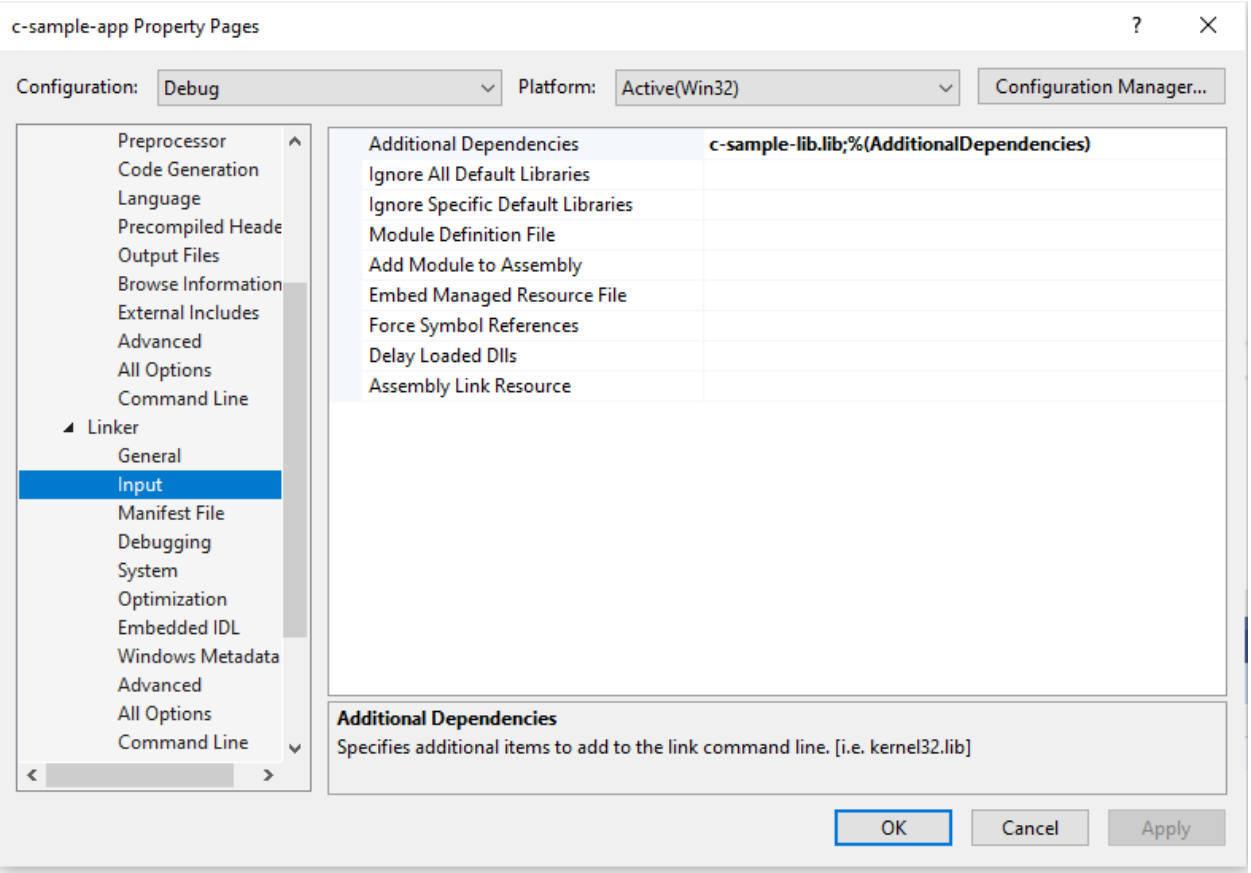

c-sample-app Property Pages
Configuration: Debug
Platform: Active(Win32)
Configuration Manager...
Configuration Properties
General
Advanced
Debugging
VC++ Directories
C/C++
Linker
Manifest Tool
XML Document Generator
Browse Information
Build Events
Custom Build Step
Code Analysis
General
Executable Directories $(VC_ExecutablePath_x86);$(CommonExecutablePath)
Include Directories $(IncludePath)
External Include Directories $(ExternalIncludePath)
Reference Directories $(VC_ReferencesPath_x86);
Library Directories ..\c-sample-lib\DebugStaticLibDeployment;$(LibraryPath)
Library WinRT Directories $(WindowsSDK_MetadataPath);
Source Directories $(VC_SourcePath);
Exclude Directories $(CommonExcludePath);$(VC_ExecutablePath_x86);$(VC_Libr
Public Project Content
Public Include Directories
All Header Files are Public No
Public C++ Module Directories
All Modules are Public No
Executable Directories
Path to use when searching for executable files while building a VC++ project. Corresponds to environment variable PATH.
OK
Cancel
Apply





If we set full path for both libraries and headers then we need to set library name for project

Linker->Input->Additional Dependencies



In this case we will compile c-sample-app and we do not need to compile c-sample-lib because we copied output files to different location and they are ready to use.

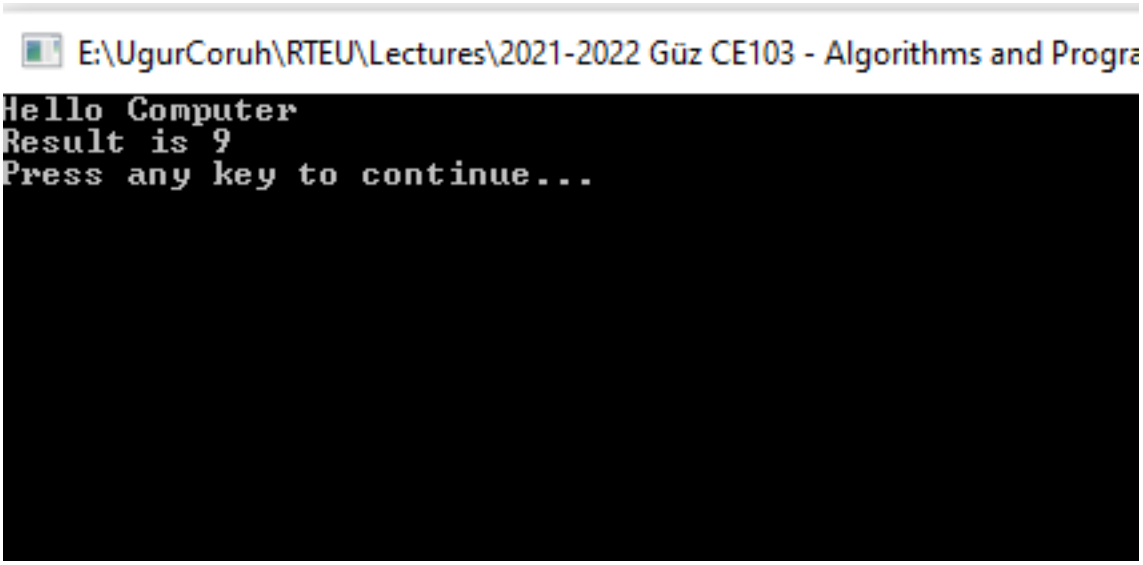
---

current source code will be like that nothing changed

```
#include <stdio.h>
#include <samplelib.h>

/// <summary>
///
/// </summary>
/// <returns></returns>
int main()
{
    int result = 0;
    //printf("Hello World!\n");
    result = sum(5, 4);
    sayHelloTo("Computer");
    printf("Result is %d \n",result);
    printf("Press any key to continue...\n");
    getchar();
    return 0;
}
```

---

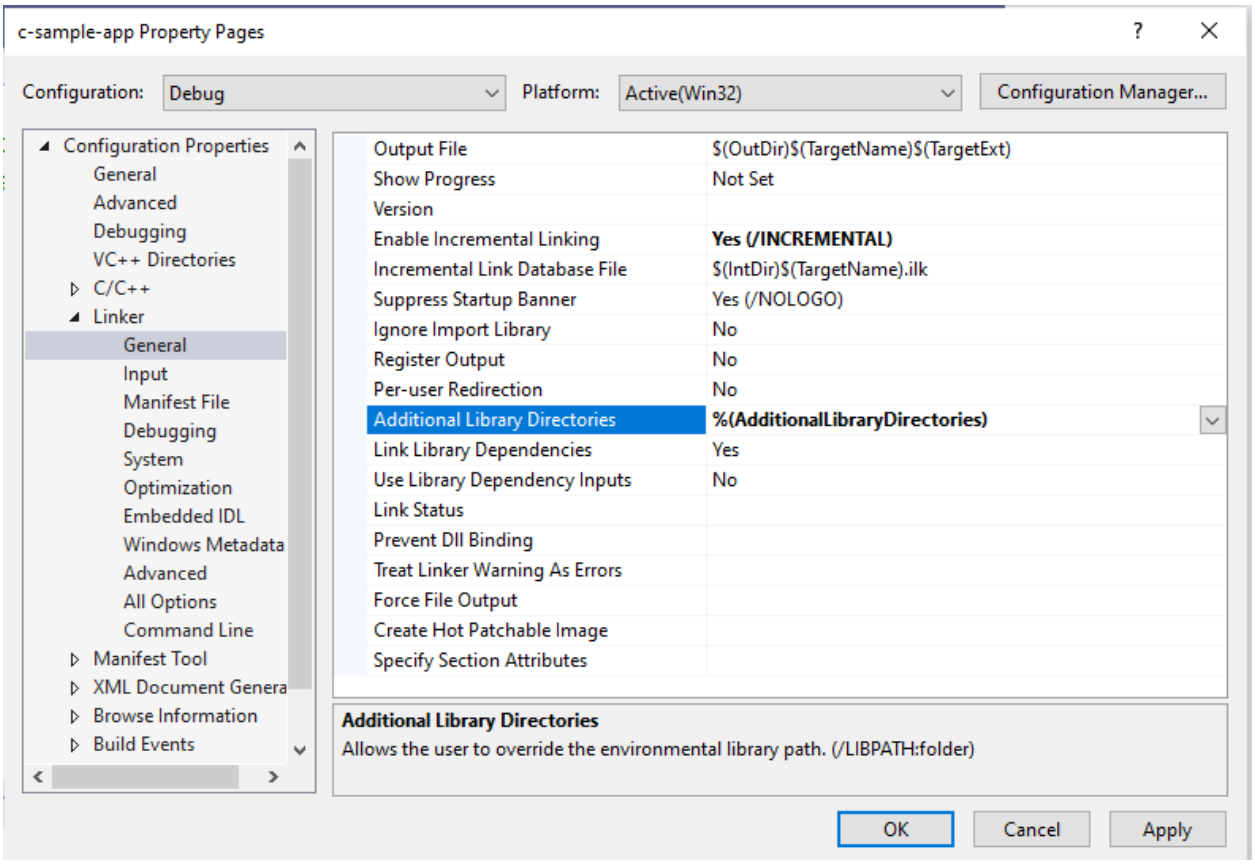
and output



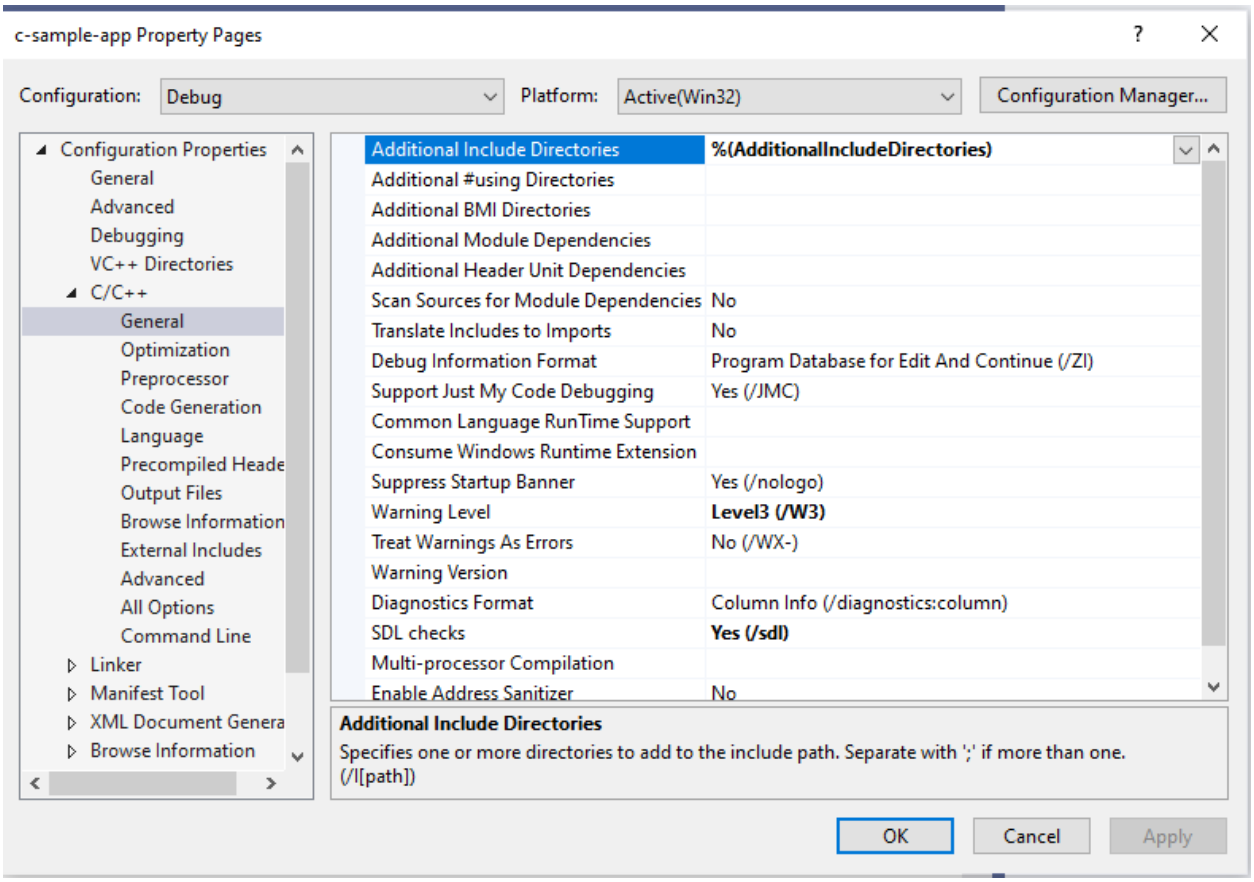
There is a option about portability that we can set for team works

We will remove all library related settings from configurations and we will write them in source code

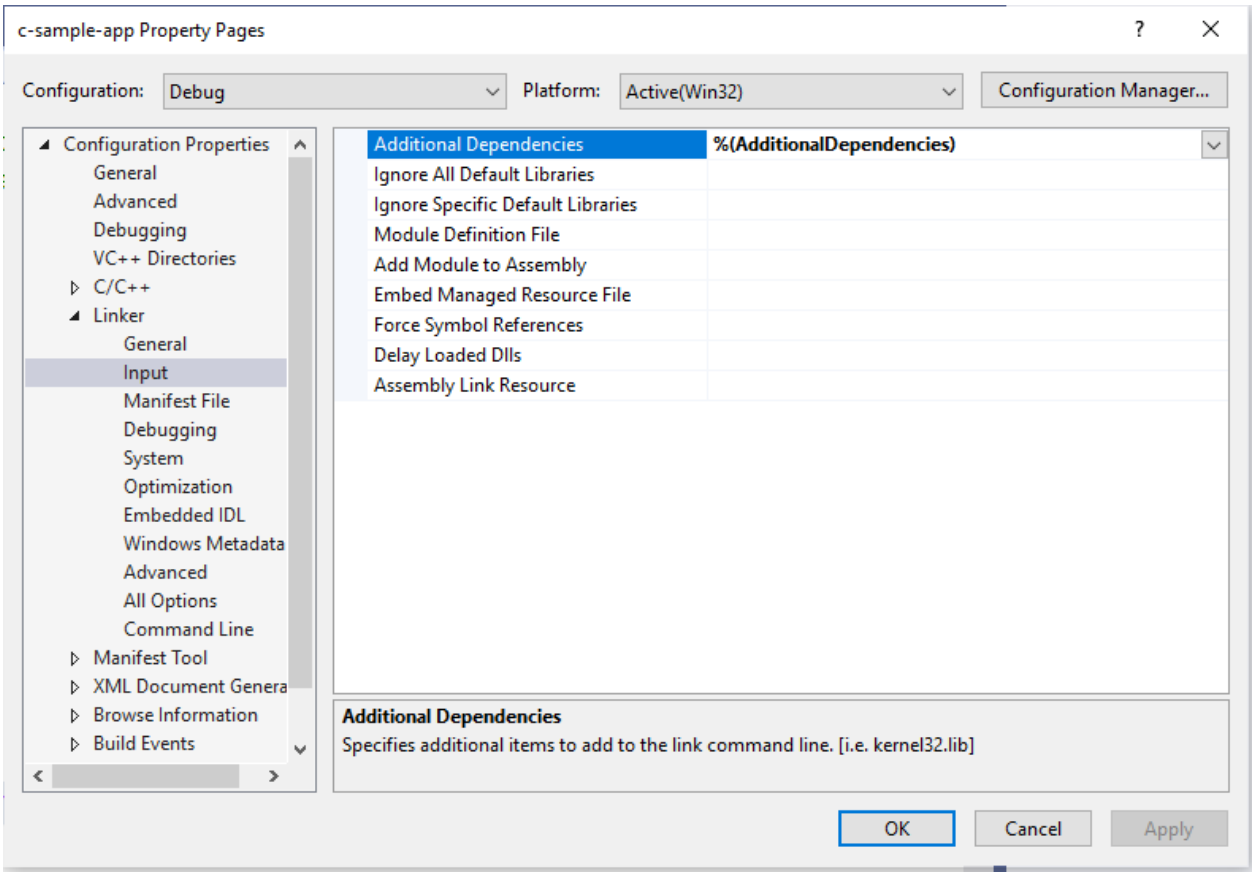
Clear linker->general->additional library directories



Clear C/C++ -> General -> Additional Include Directories



Clear Linker->Input->Additional Dependencies



Now we can set this configurations in source code as follow

```
#pragma comment(lib, "..\\DebugStaticLibDeployment\\c-sample-lib.lib")
#include "..\DebugStaticLibDeployment\samplelib.h"

#include <stdio.h>

/// <summary>
///
/// </summary>
/// <returns></returns>
int main()
{
    int result = 0;
    //printf("Hello World!\n");
    result = sum(5, 4);
    sayHelloTo("Computer");
    printf("Result is %d \n",result);
    printf("Press any key to continue...\n");
    getchar();
    return 0;
}
```

with this configuration if your friends download this code then they can run them with their environment without setting a path.

0.4.2 C++ Programming (Static Library)

0.4.2.1 Visual Studio Community Edition All steps are similar with C programming above, but you do not need to delete pch.h

You should take care about compiled source codes

for example if your code is compiled for x86 then your application also should use the x86 configuration else x64 then library should be x64 complied version.

Source will look like the following

```
// cpp-sample-app.cpp : This file contains the 'main' function. Program execution begins and ends there
//

#pragma comment(lib, "..\\DebugStaticLibDeployment\\cpp-sample-lib.lib")

#include "..\DebugStaticLibDeployment\samplelib.h"

#include <iostream>

int main()
{
    std::cout << "Hello World!\n";

    int result = 0;
    //printf("Hello World!\n");
    result = sum(5, 4);
    sayHelloTo("Computer");
    printf("Result is %d \n", result);
    printf("Press any key to continue...\n");
    getchar();
    return 0;

}
```

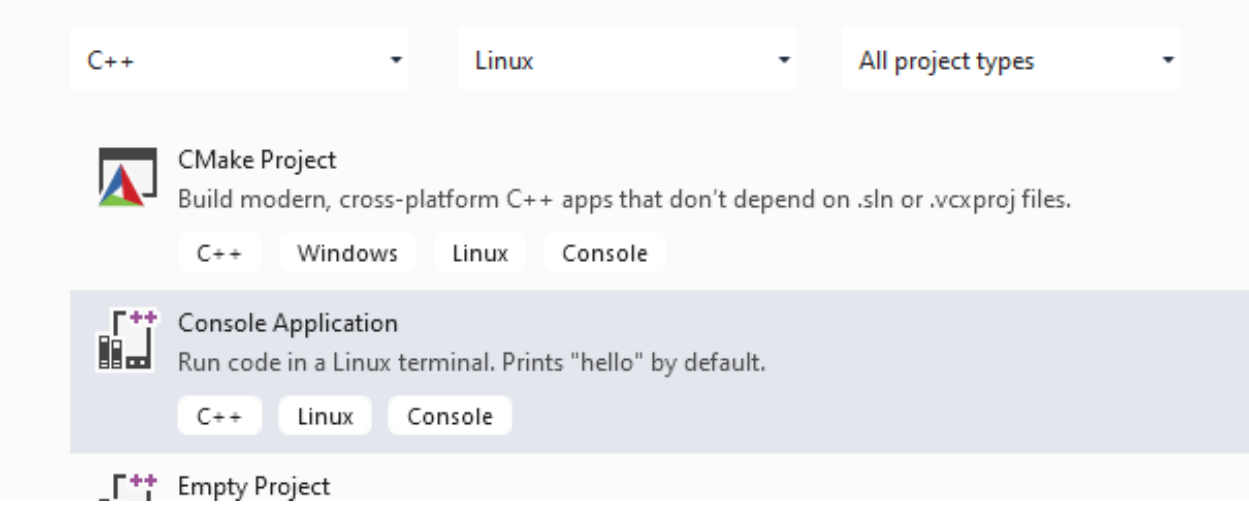
0.4.3 C/C++ WSL Option

Install WSL

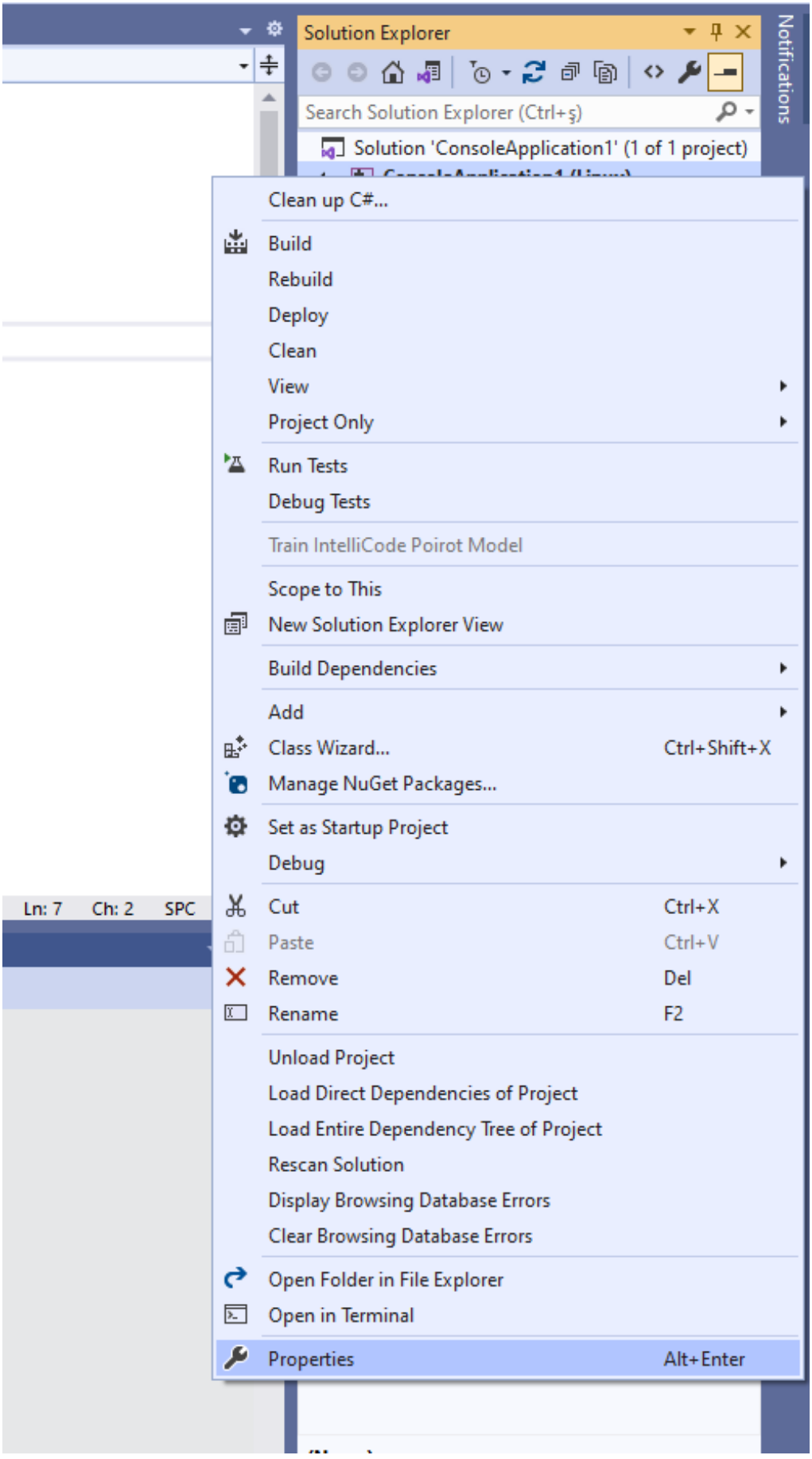
GitHub - ucoruh/ns3-wsl-win10-setup: ns3 windows 10 WSL2 setup and usage[5]

Create a Linux project

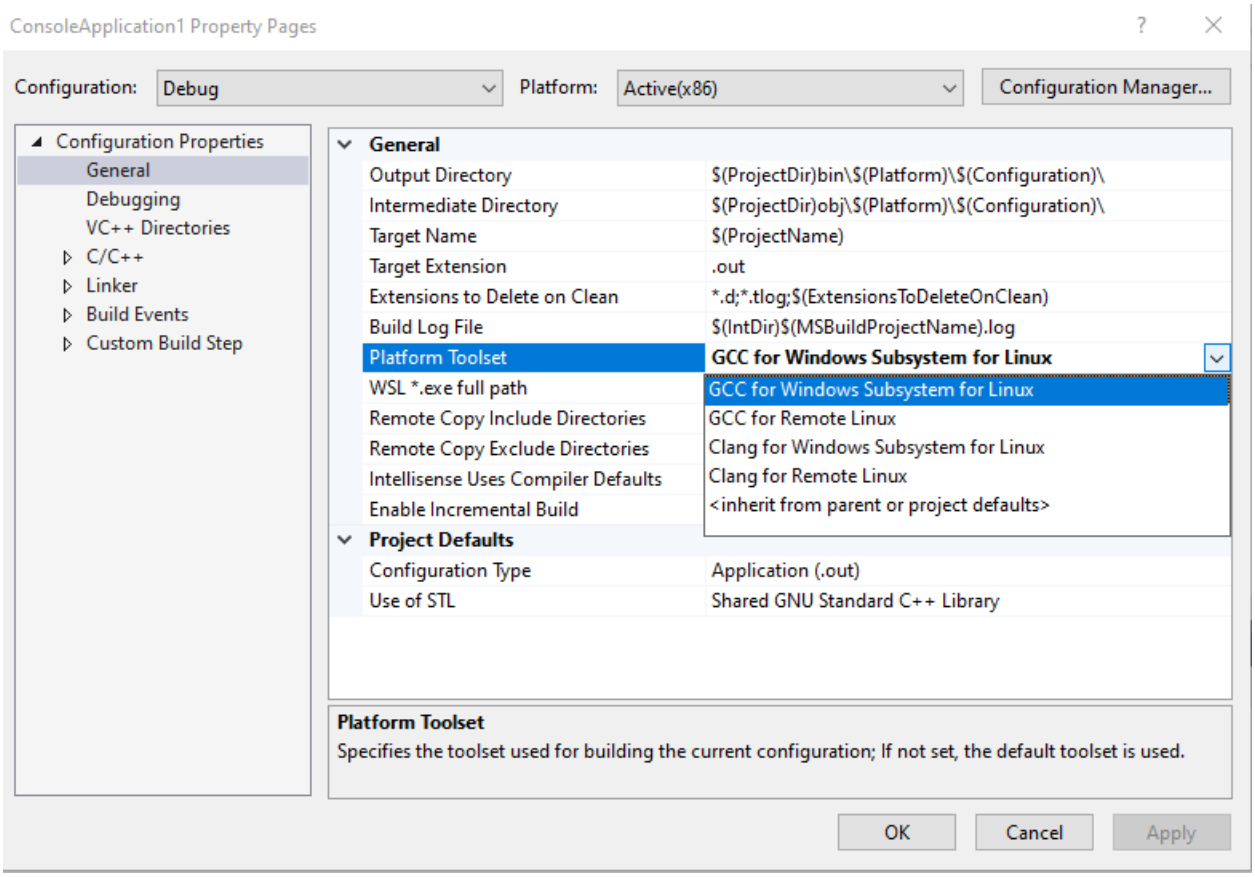
[5] https://github.com/ucoruh/ns3-wsl-win10-setup



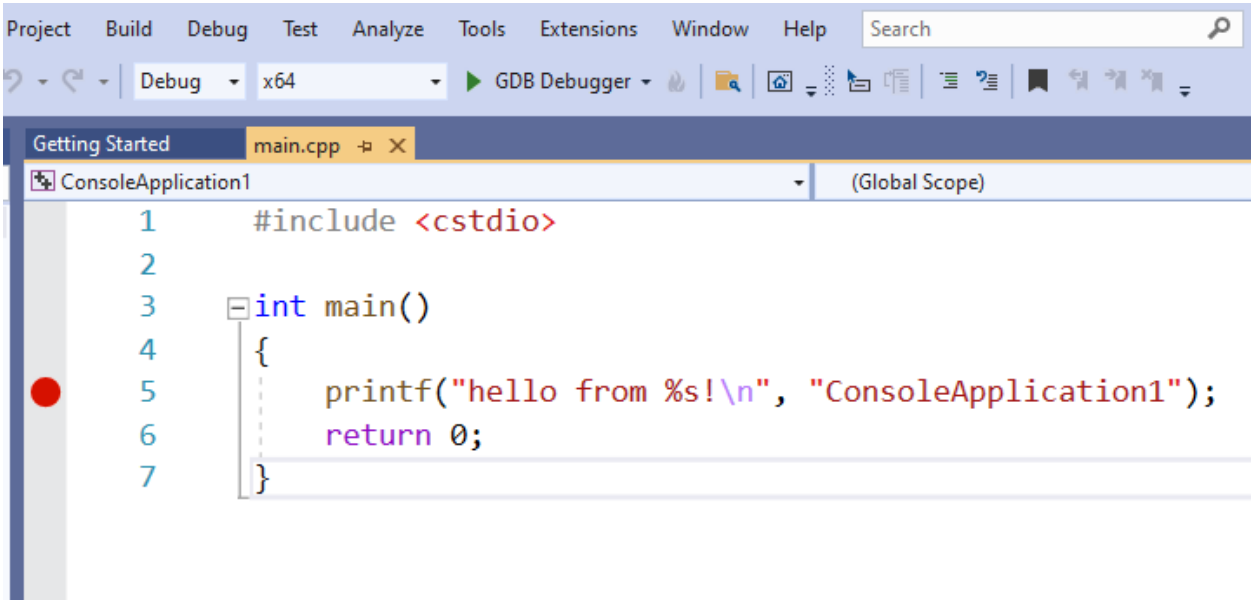
Configure Platform Toolset to WSL


Solution Explorer
Search Solution Explorer (Ctrl+ş)
Solution 'ConsoleApplication1' (1 of 1 project)
Notifications
Clean up C#...
Build
Rebuild
Deploy
Clean
View
Project Only
Run Tests
Debug Tests
Train IntelliCode Poirot Model
Scope to This
New Solution Explorer View
Build Dependencies
Add
Class Wizard... Ctrl+Shift+X
Manage NuGet Packages...
Set as Startup Project
Debug
Cut Ctrl+X
Paste Ctrl+V
Remove Del
Rename F2
Unload Project
Load Direct Dependencies of Project
Load Entire Dependency Tree of Project
Rescan Solution
Display Browsing Database Errors
Clear Browsing Database Errors
Open Folder in File Explorer
Open in Terminal
Properties Alt+Enter
Ln: 7 Ch: 2 SPC

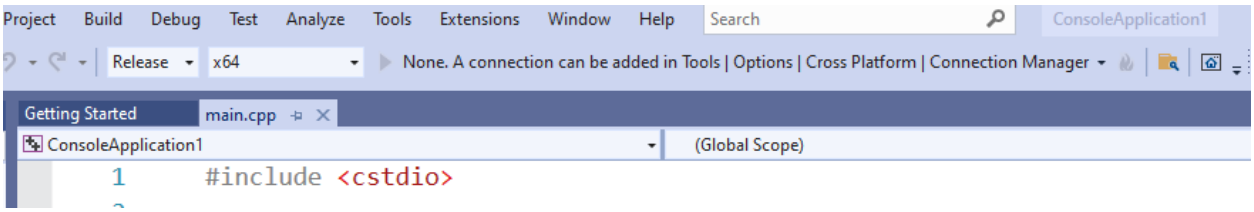
Select GCC for Windows Subsystem for Linux



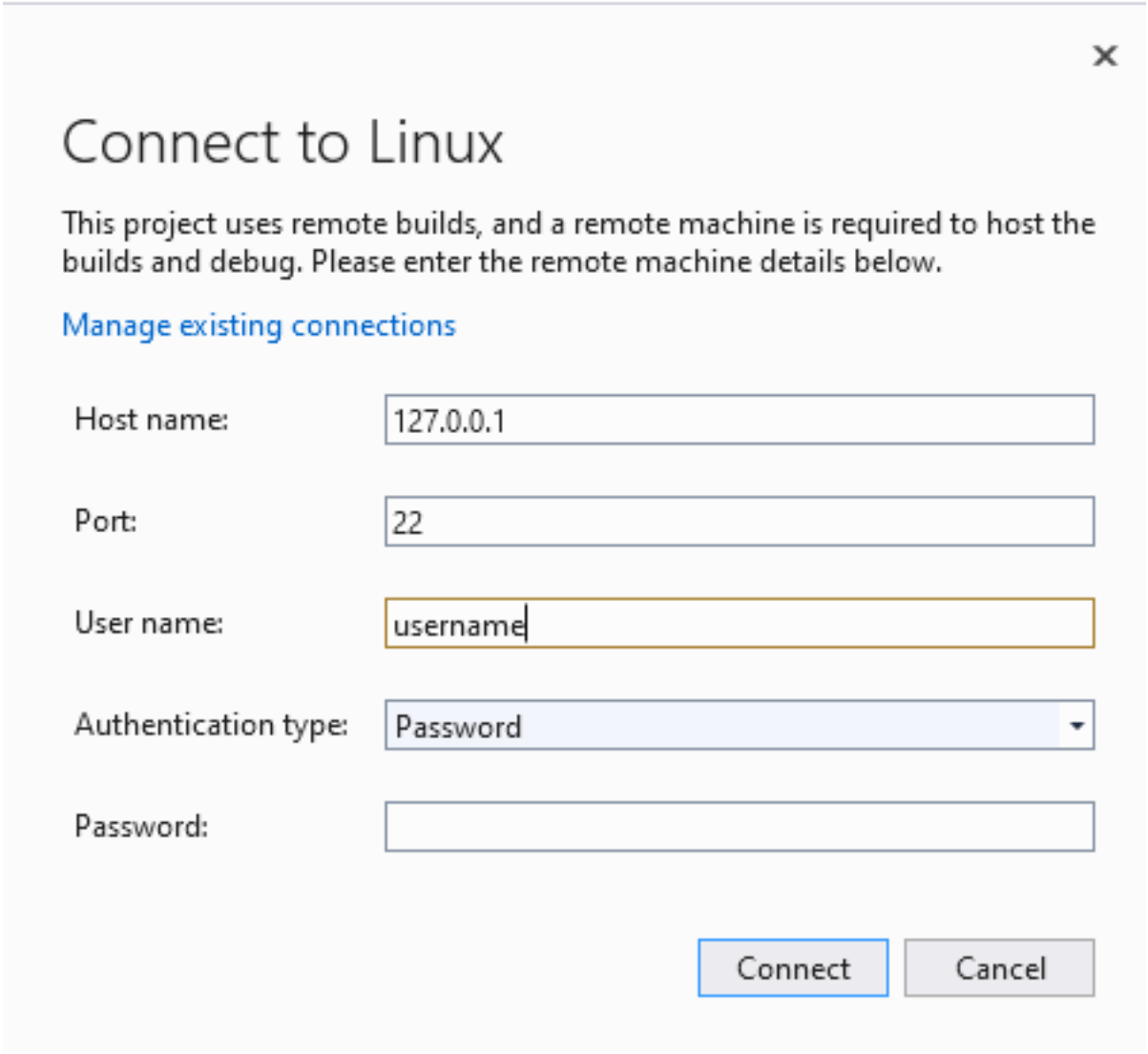
Put a breakpoint and run debugger



In the debugger for WSL you can use local WSL installation but if you want to run it on Release setting it require a SSH connection.



Configure SSH parameters



so you have to complete the following steps.

0.4.4 C/C++ Remote Linux Option over SSH

Enable SSH

SSH on Windows Subsystem for Linux (WSL) | Illuminia Studios[6]

Connect to Remote WSL Environment

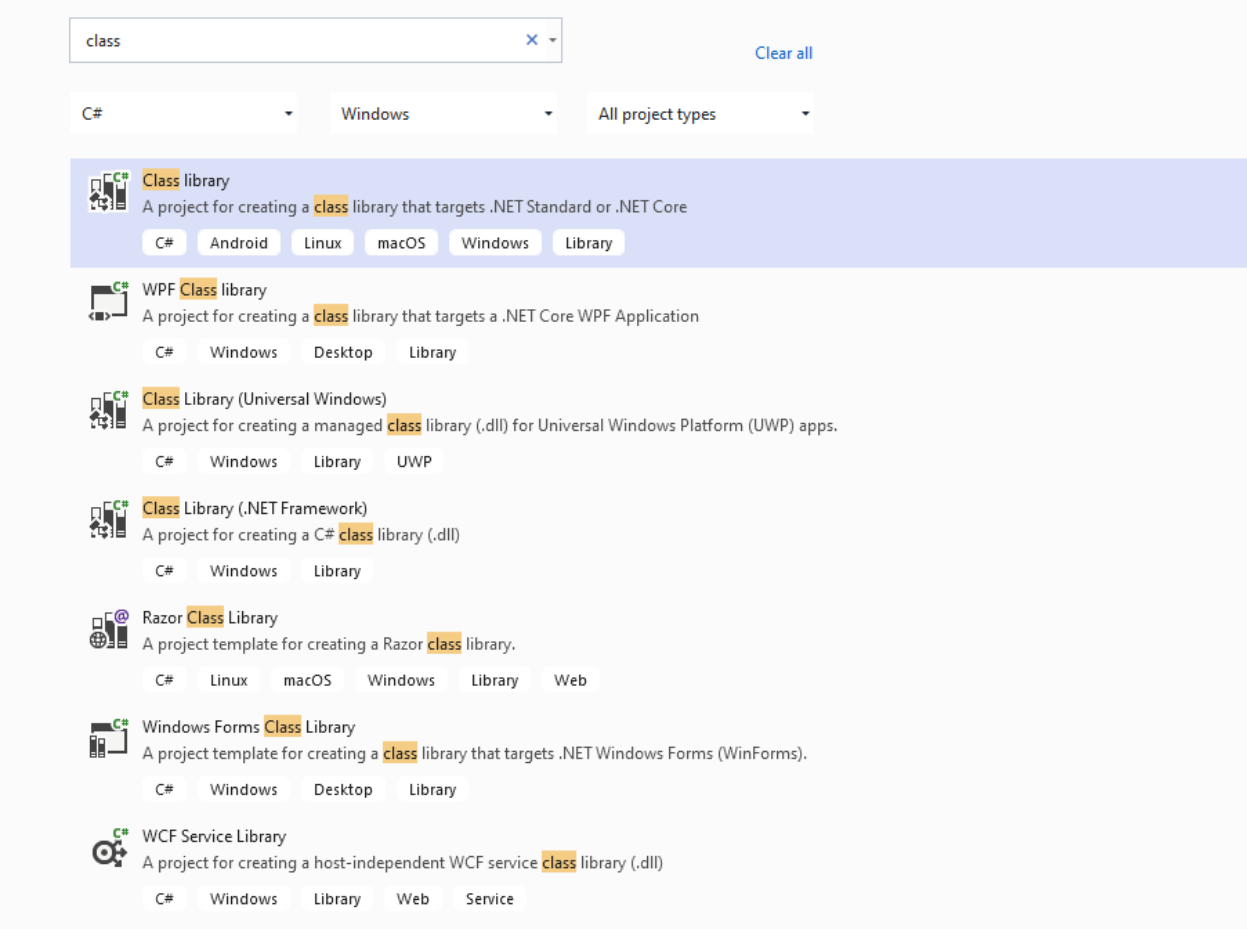
[6] https://www.illuminiastudios.com/dev-diaries/ssh-on-windows-subsystem-for-linux/

Bağlan hedef Linux sisteminize Visual Studio | Microsoft Docs[7]

0.4.5 C# Programming (Dinamik Library)

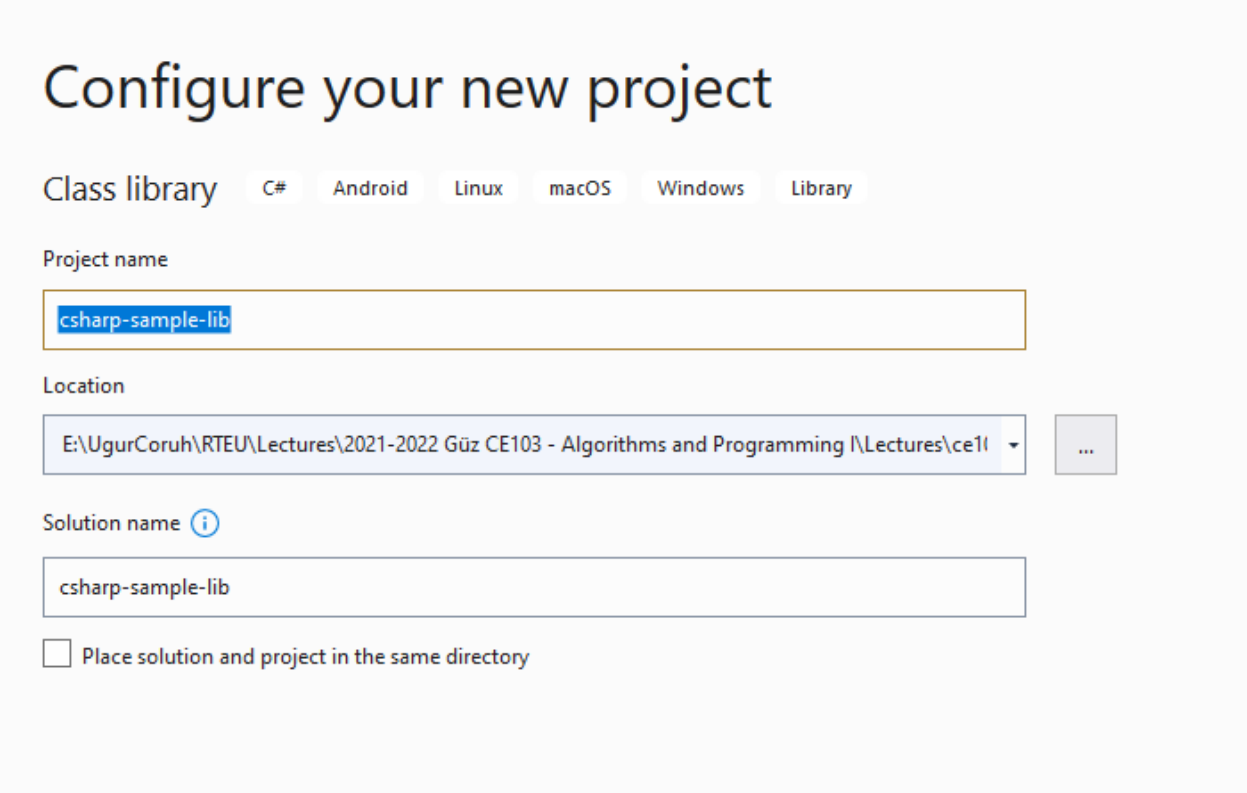
0.4.5.1 Visual Studio Community Edition In C# project we will create class library we have several options

for this sample we will select .NET core that we can build cross platform library

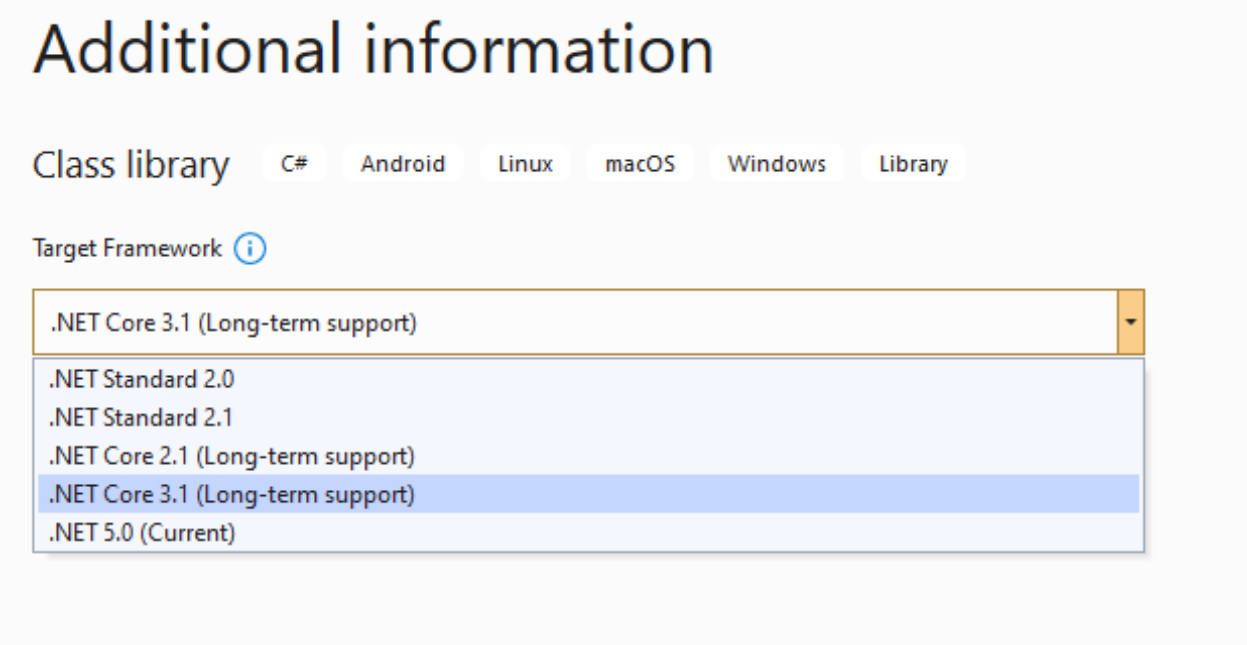


There is no static library option

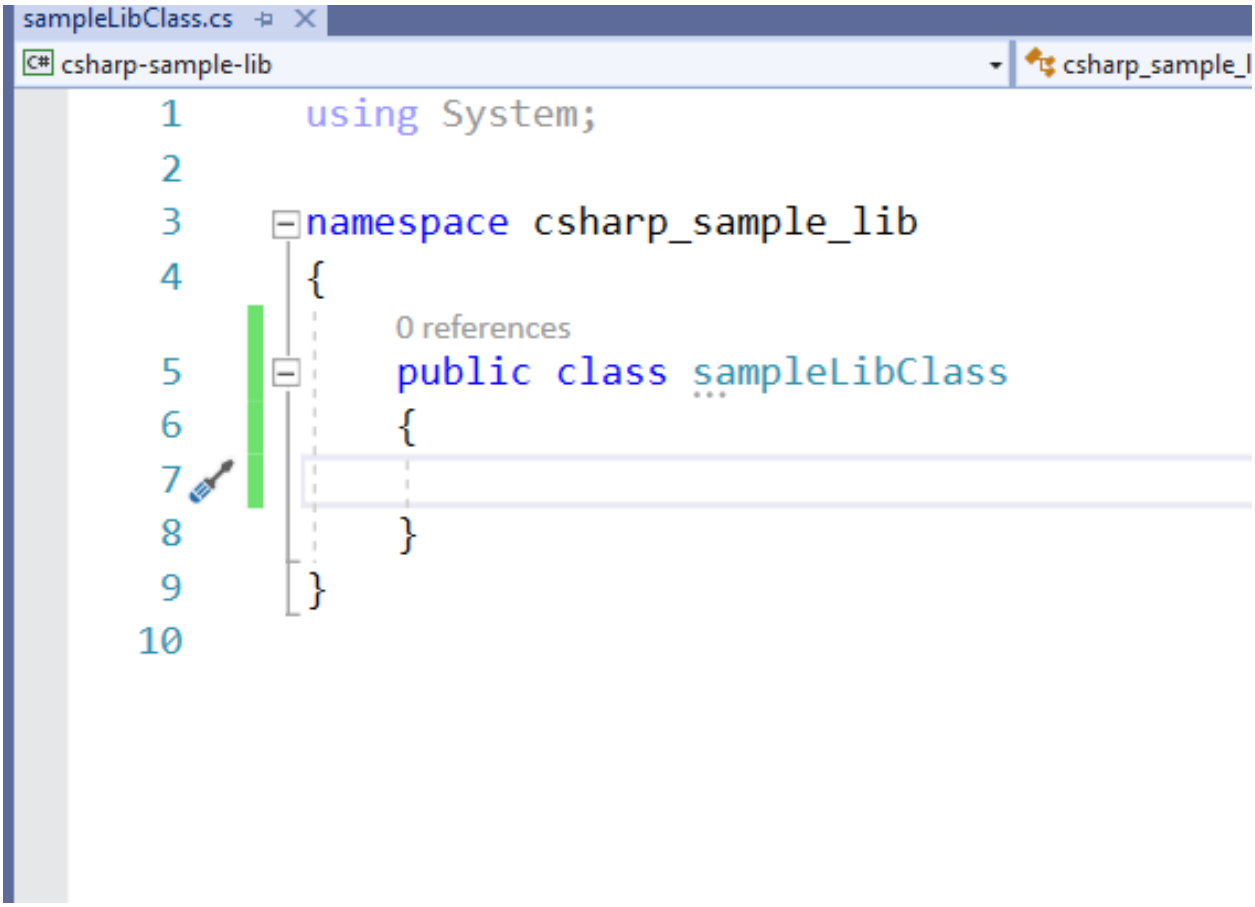
[7] https://docs.microsoft.com/tr-tr/cpp/linux/connect-to-your-remote-linux-computer?view=msvc-160



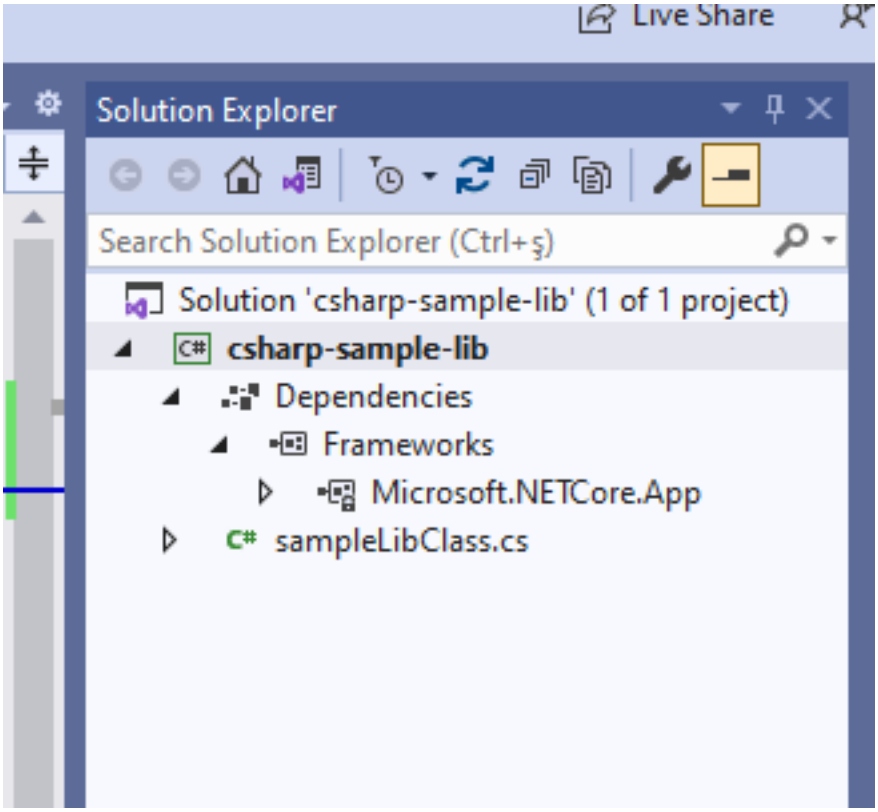
We will select .Net Core 3.1



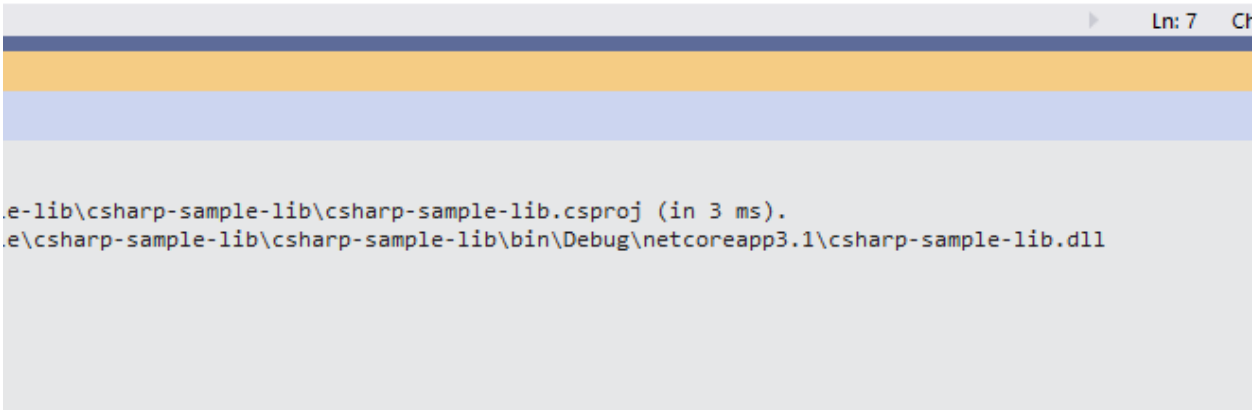
You will have default empty class library file



In the project you can see .NETcore reference

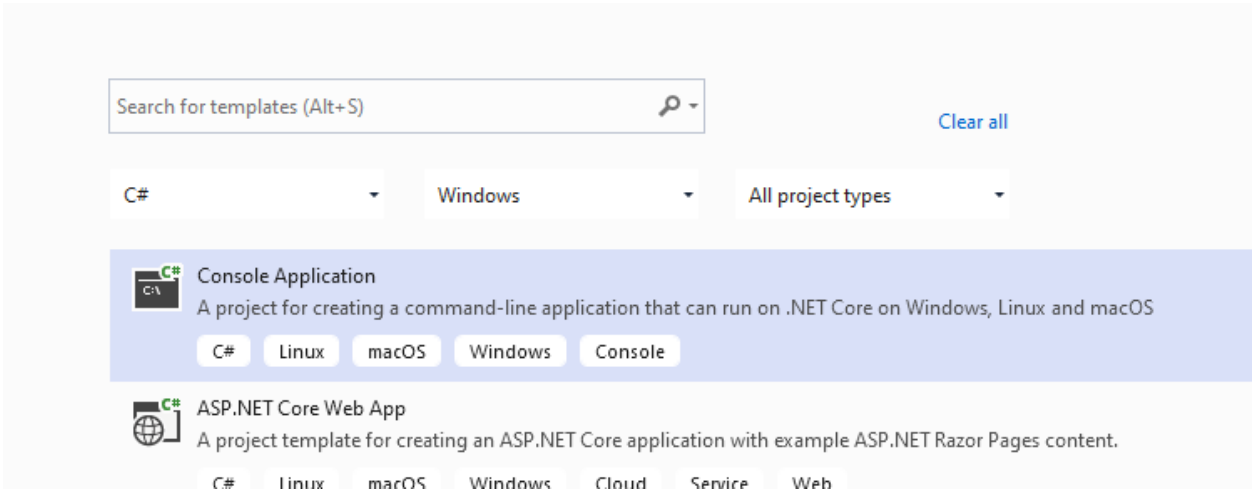


We can build empty class library that generate dll for our application

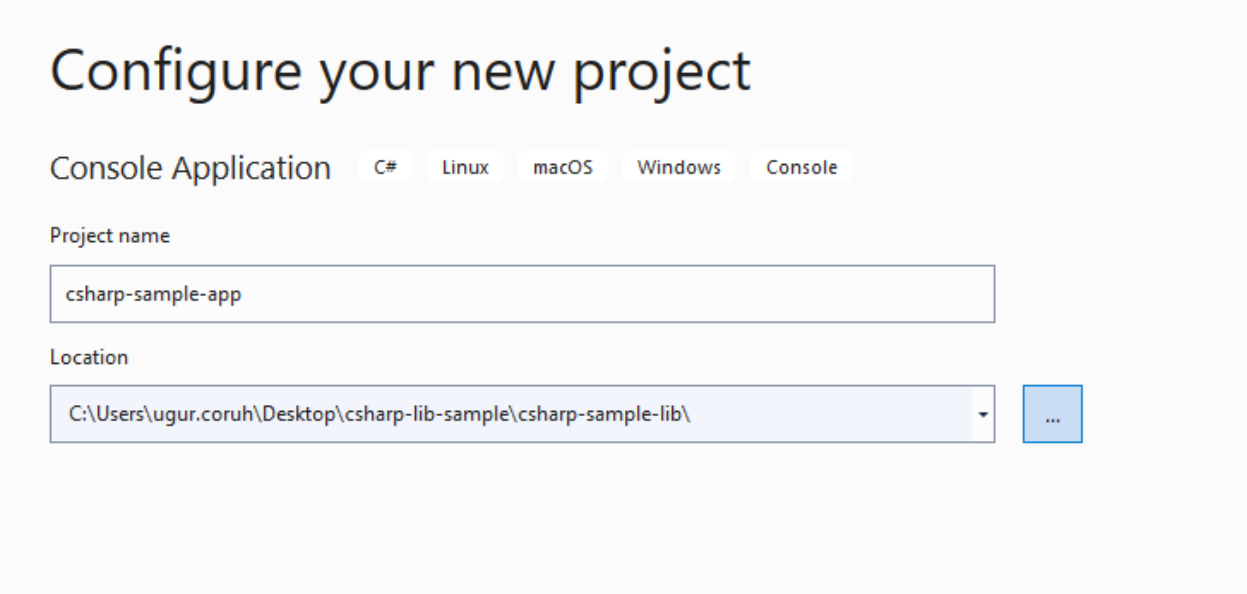


Now we will add Console Application but this will also use .NETCore

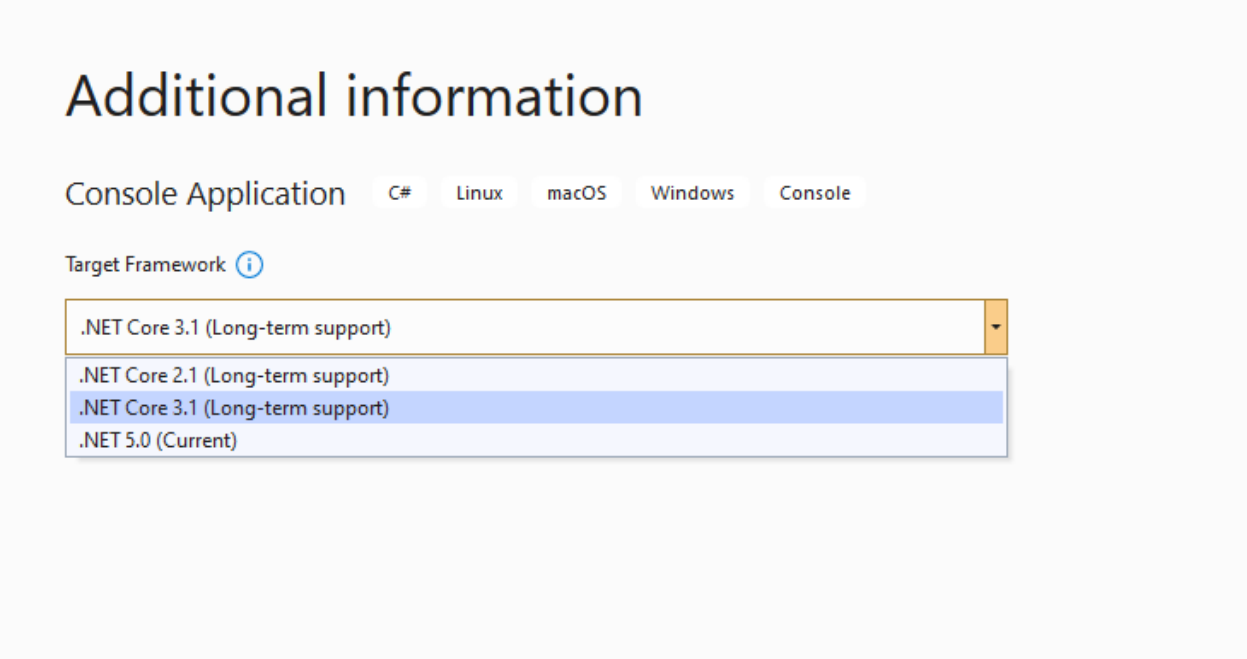
Select New Project



Name the project



Select .NETCore framework

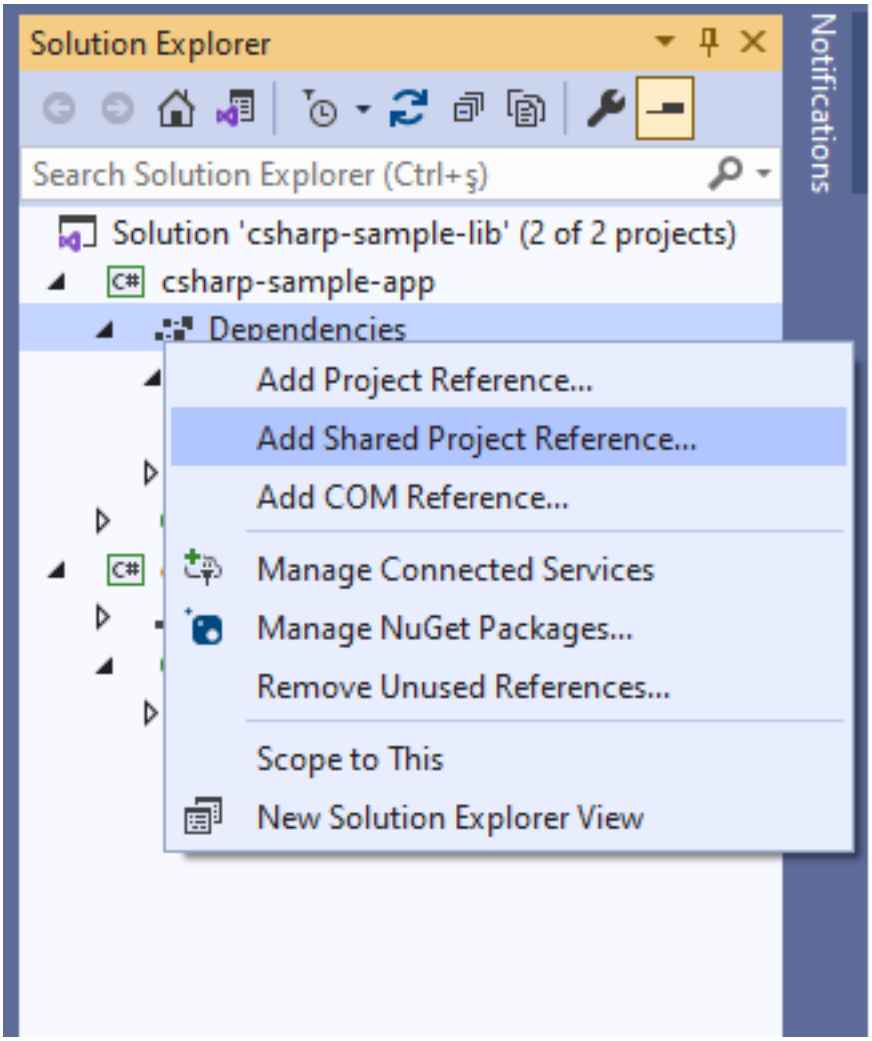


You will have the following sample main.cs file

```
using System;

namespace csharp_sample_app
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Hello World!");
        }
```

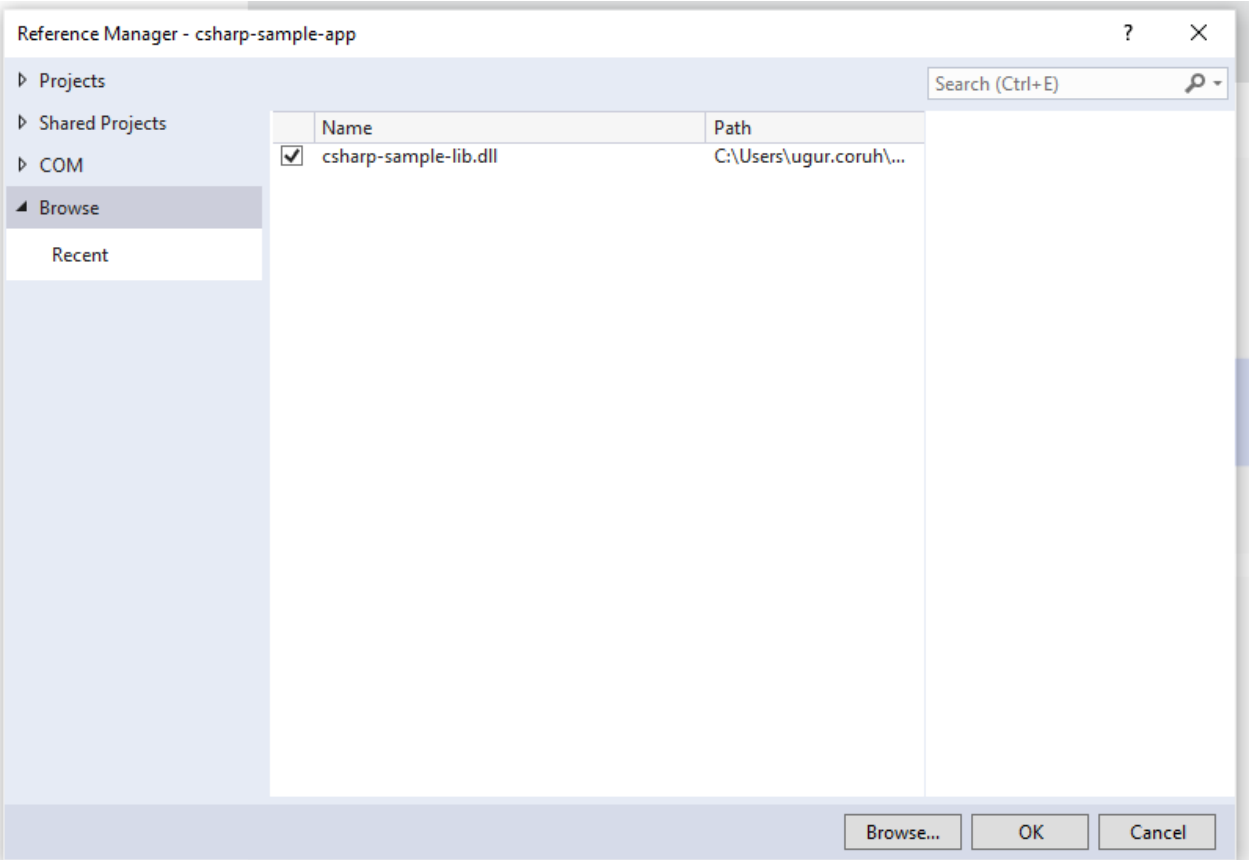
```
    }
}
```

Now we can link projects with adding references open reference section



---

browse for class library project output folder and select output dll file for console application



---

now we can update our library code and use it in console application

copy following sample to sampleLibClass file in the library

```
using System;

namespace csharp_sample_lib
{
    public class sampleLibClass
    {
        public static void sayHelloTo(string name)
        {
            if (!String.IsNullOrEmpty(name))
            {
                Console.WriteLine("Hello " + name);
            }
            else
            {
                Console.WriteLine("Hello There");
            }

        }

        public static int sum(int a, int b)
        {
            int c = 0;
            c = a + b;
            return c;
        }
```

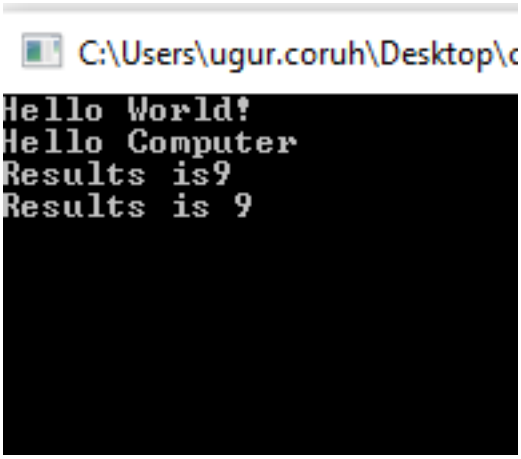
```
    }
}
```

---

after this operation copy following sample to console application and build app then you can run

```
using csharp_sample_lib;
using System;

namespace csharp_sample_app
{
    class Program
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Hello World!");

            sampleLibClass.sayHelloTo("Computer");
            int result = sampleLibClass.sum(5, 4);
            Console.WriteLine("Results is" + result);
            Console.WriteLine("Results is {0}", result);
            Console.Read();
        }
    }
}
```

You will see following output that mean we called DLL functions



Also we can publish this console application with dll for linux environment or others

for linux environment we should install .NETCore

---

follow the link below or commands that I shared with you as below for deployment

How to Install Dotnet Core on Ubuntu 20.04 – TecAdmin[8]

Step 1 – Enable Microsoft PPA

```
 wget https://packages.microsoft.com/config/ubuntu/20.04/packages-microsoft-prod.deb
sudo dpkg -i packages-microsoft-prod.deb
```

[8] https://tecadmin.net/how-to-install-net-core-on-ubuntu-20-04/

Step 2 – Installing Dotnet Core SDK

```
sudo apt update
sudo apt install apt-transport-https
sudo apt install dotnet-sdk-3.1
```

Step 3 – Install Dotnet Core Runtime Only

To install .NET Core Runtime on Ubuntu 20.04 LTS system, execute the commands:

```
sudo apt update
```

To install the previous version of .Net core runtime 2.1, type:

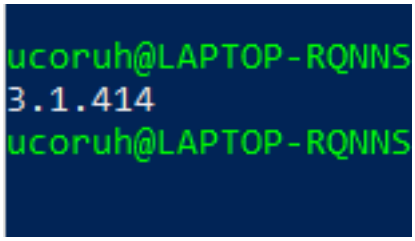
```
sudo apt install dotnet-runtime-2.1
```

Press “y” for any input prompted by the installer.

Step 4 – (Optional) Check .NET Core Version

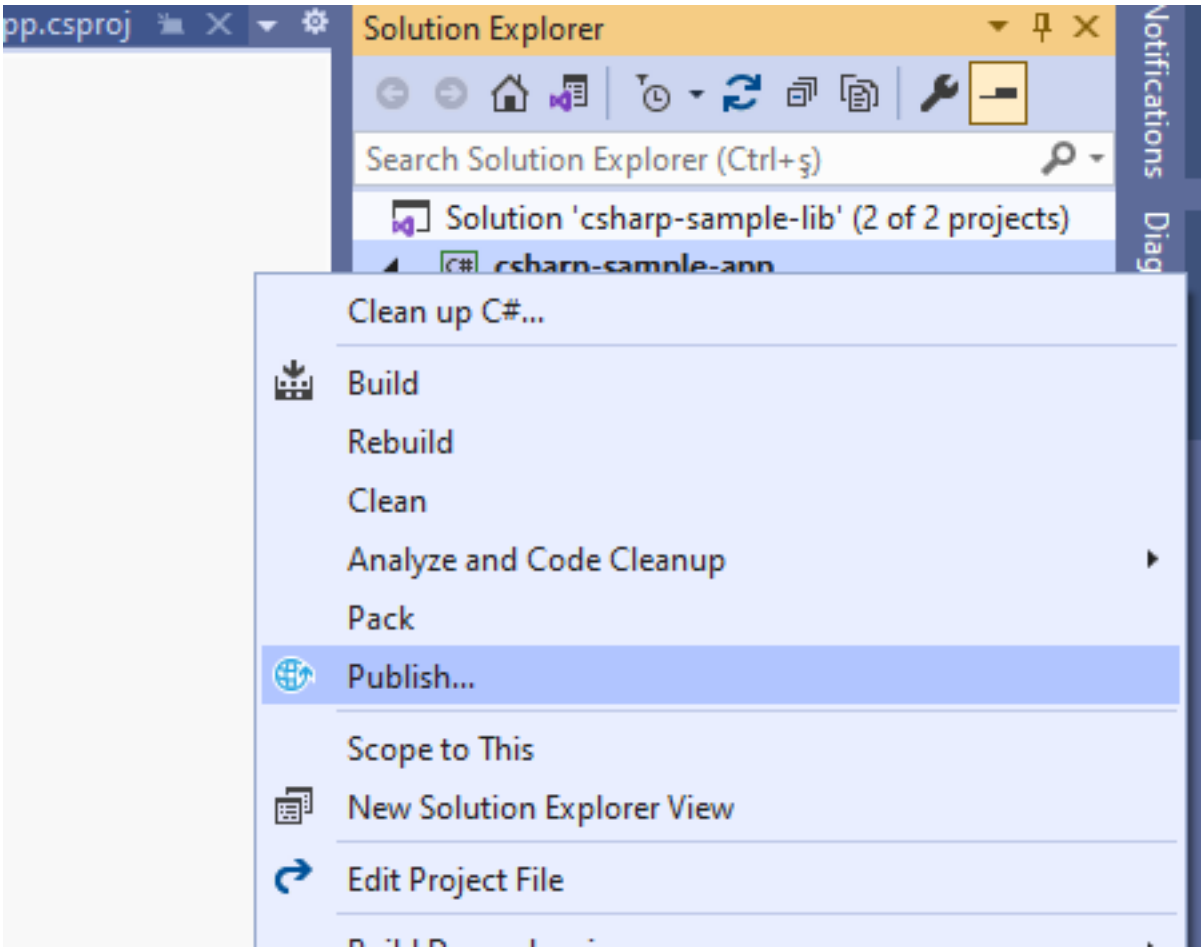
You can use dotnet command line utility to check installed version of .NET Core on your system. To check dotnet version, type:

```
dotnet --version
```

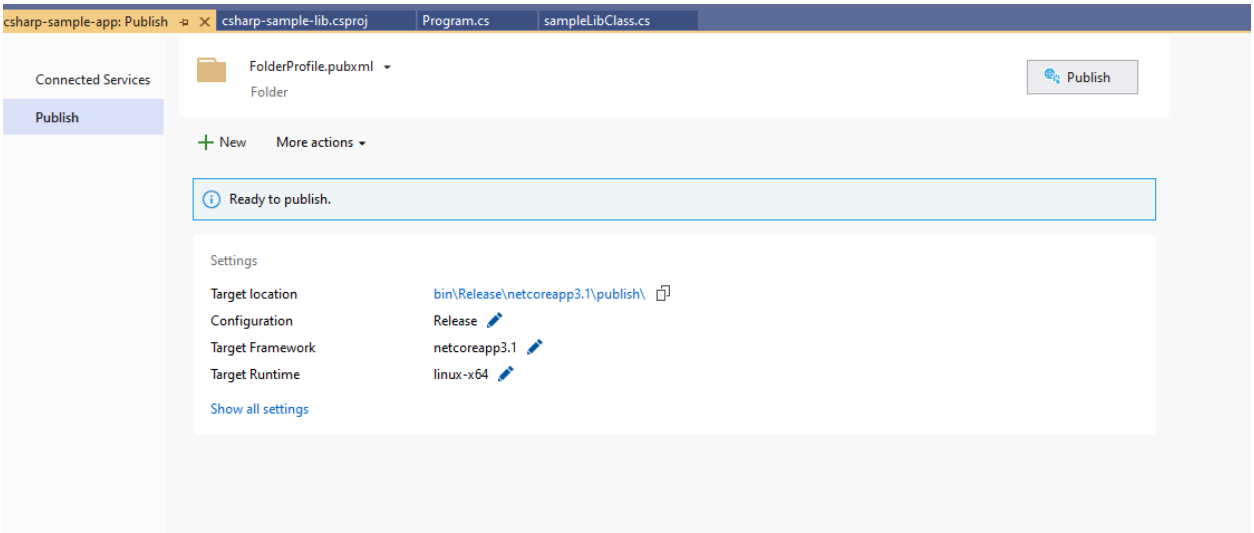


Now we will publish our application as single executable

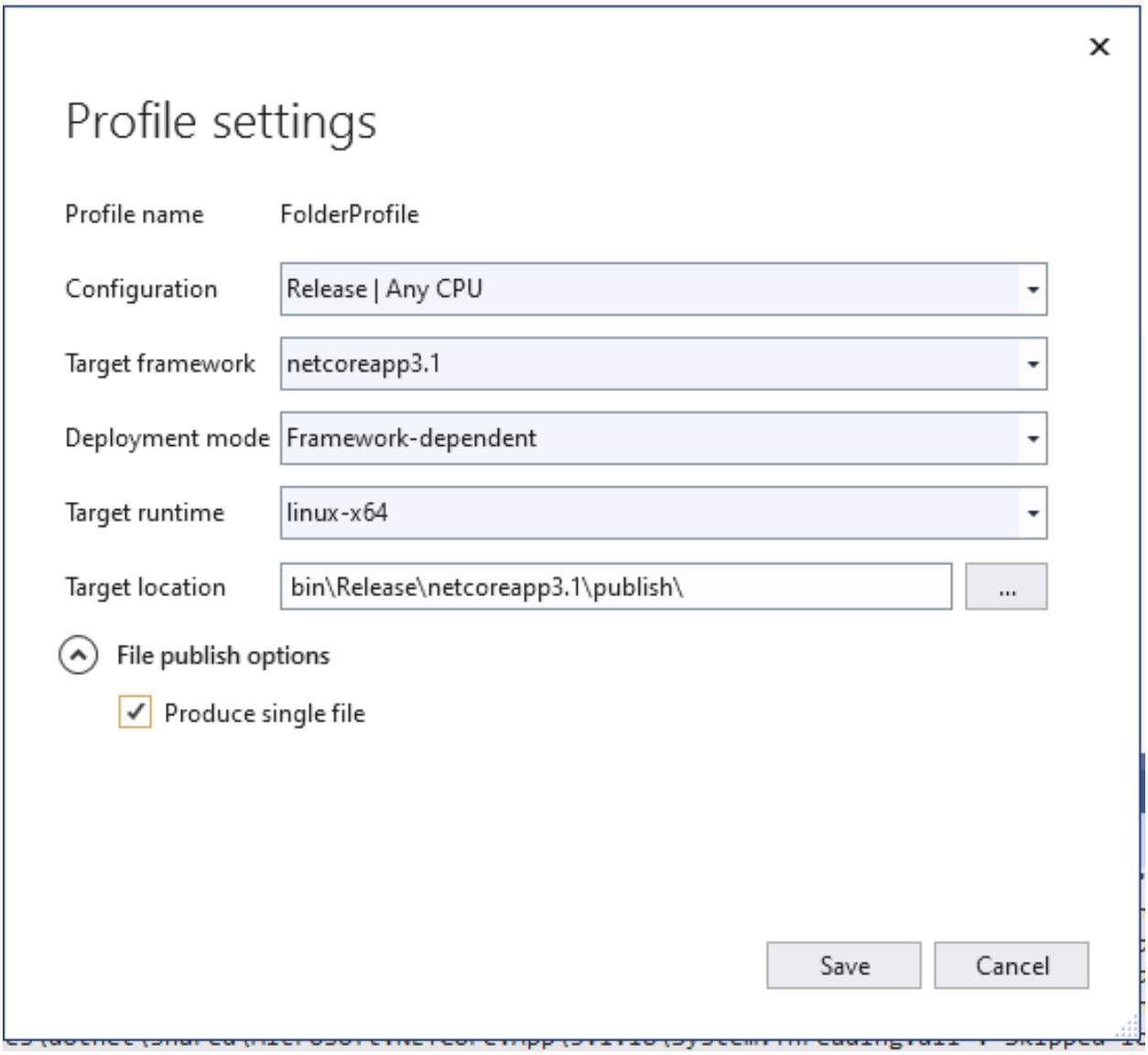
Open publish menu



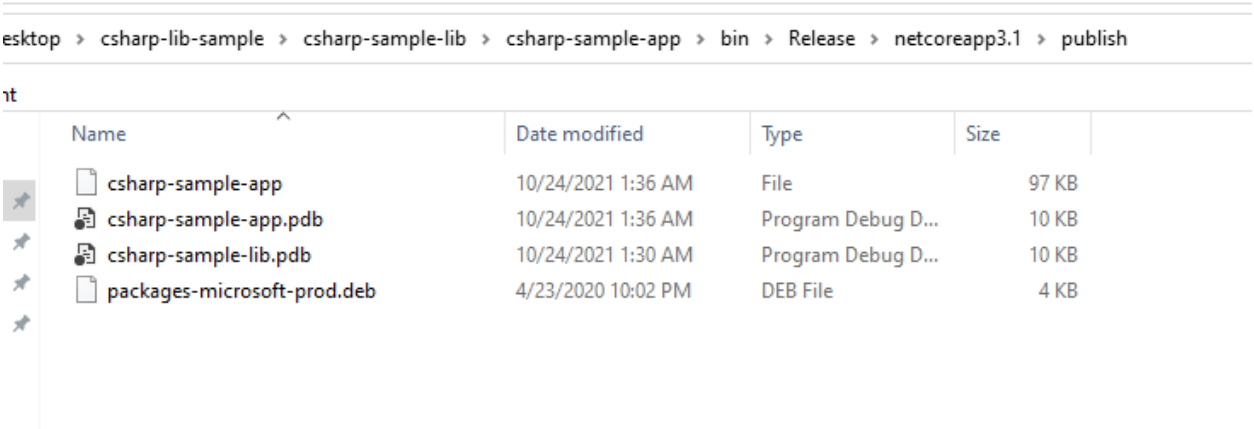
Select netcoreapp3.1 and Release for linux-x64



Select produce single file

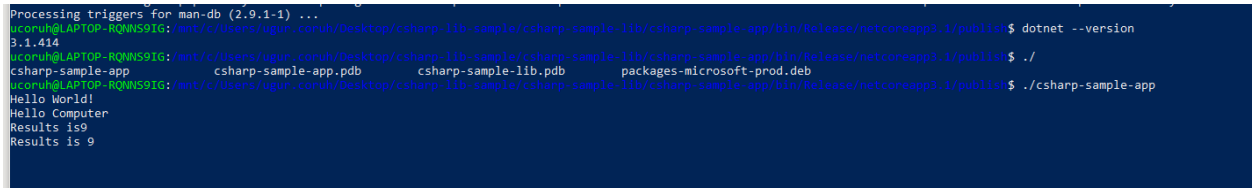


After succesfull publish you will have linux binary that you can run with WSL



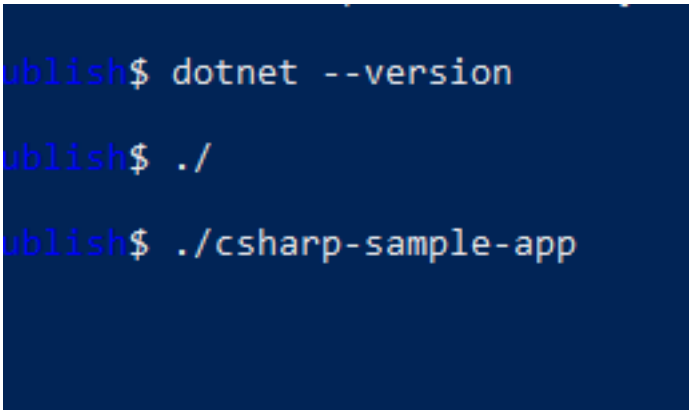
Open WSL and enter the path where this folder located

and run application as follow

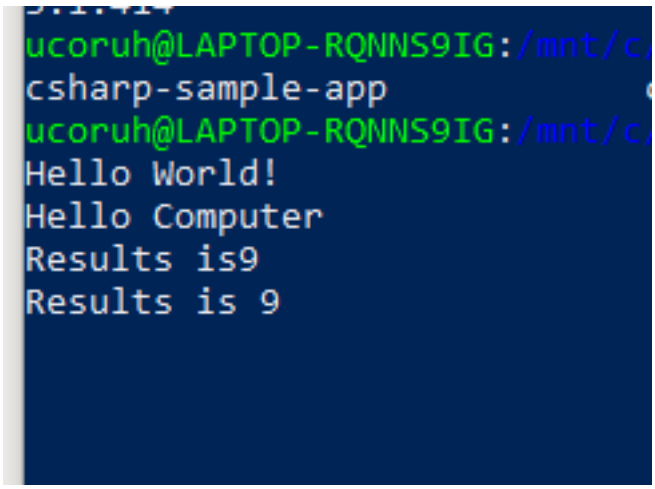


---

check dotnet –version and then run application

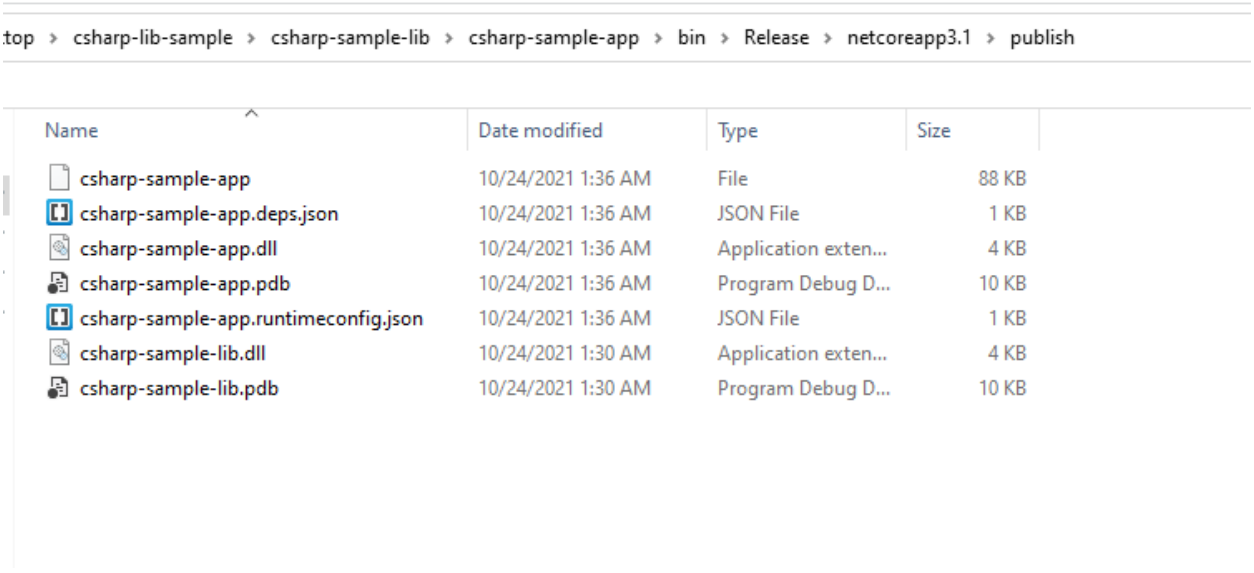


you will see similar output



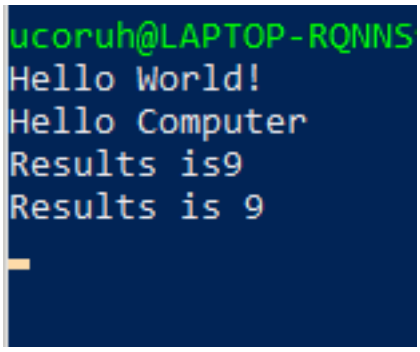
In this sample we created single application from settings lets try with shared library located option uncheck the “produce single file” option and publish again.

Then you will have the following outputs



If you run csharp-sample-app

you will have the same output



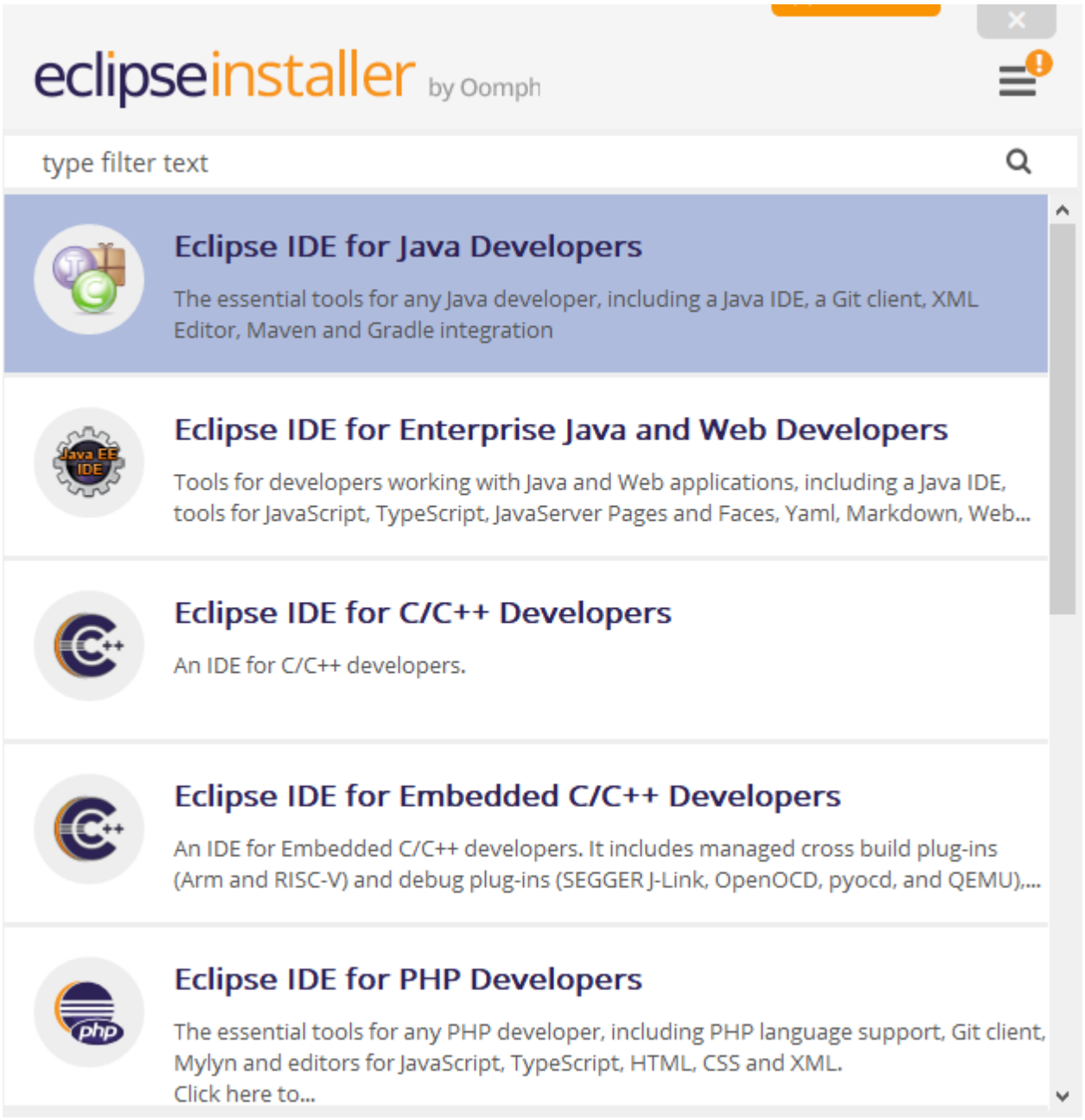
0.4.6 Java Programming

0.4.6.1 Eclipse IDE You should download and install eclipse installer and then you should select Eclipse IDE for Java Developers

Eclipse Installer 2021-09 R | Eclipse Packages[9]

[9] https://www.eclipse.org/downloads/packages/installer

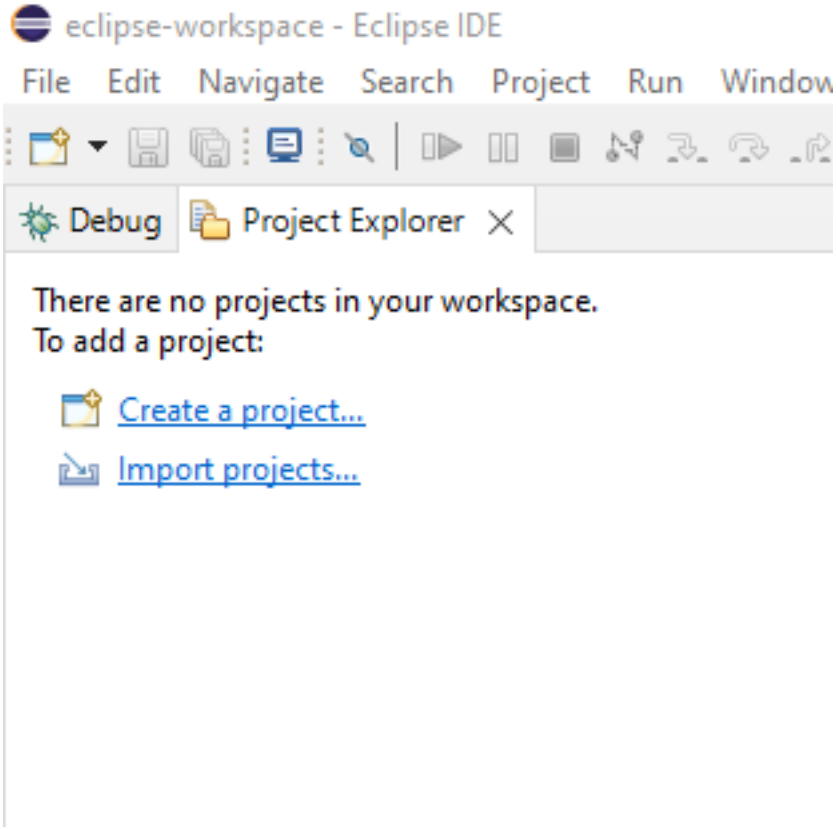

eclipseinstaller
by Oomph


eclipseinstaller by Oomph
type filter text
Eclipse IDE for Java Developers
The essential tools for any Java developer, including a Java IDE, a Git client, XML Editor, Maven and Gradle integration
Eclipse IDE for Enterprise Java and Web Developers
Tools for developers working with Java and Web applications, including a Java IDE, tools for JavaScript, TypeScript, JavaServer Pages and Faces, Yaml, Markdown, Web...
Eclipse IDE for C/C++ Developers
An IDE for C/C++ developers.
Eclipse IDE for Embedded C/C++ Developers
An IDE for Embedded C/C++ developers. It includes managed cross build plug-ins (Arm and RISC-V) and debug plug-ins (SEGGER J-Link, OpenOCD, pyocd, and QEMU),...
Eclipse IDE for PHP Developers
The essential tools for any PHP developer, including PHP language support, Git client, Mylyn and editors for JavaScript, TypeScript, HTML, CSS and XML.
Click here to...


DONATE
eclipseinstaller by Oomph
Eclipse IDE for Java Developers
details
The essential tools for any Java developer, including a Java IDE, a Git client, XML Editor, Maven and Gradle integration.
Java 11+ VM
C:\Program Files\Java\jdk-16.0.1
Installation Folder
C:\Users\ugur.coruh\eclipse\java-2021-09
create start menu entry
create desktop shortcut
INSTALLING
Cancel Installation
BACK

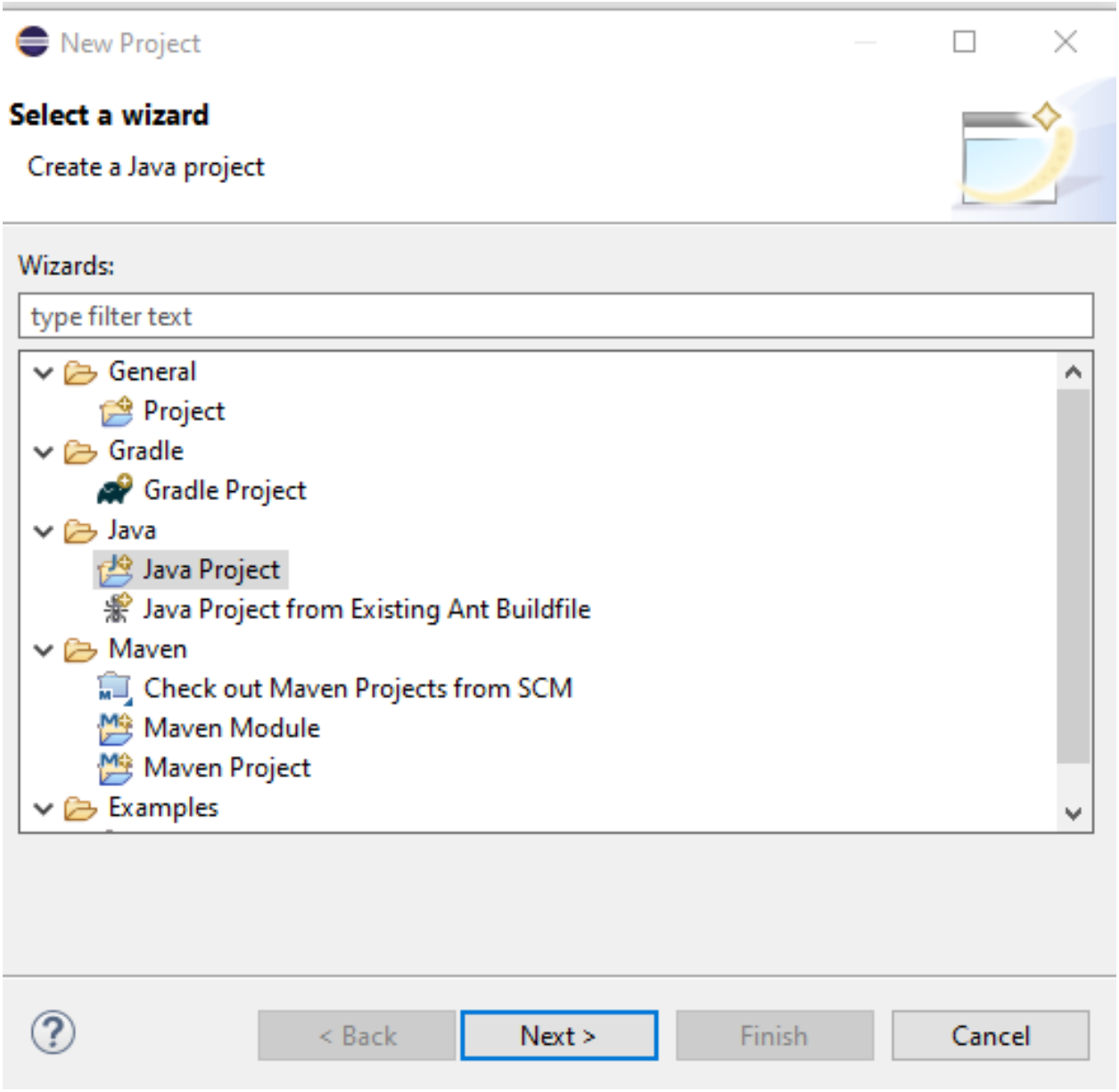
---

select create a project

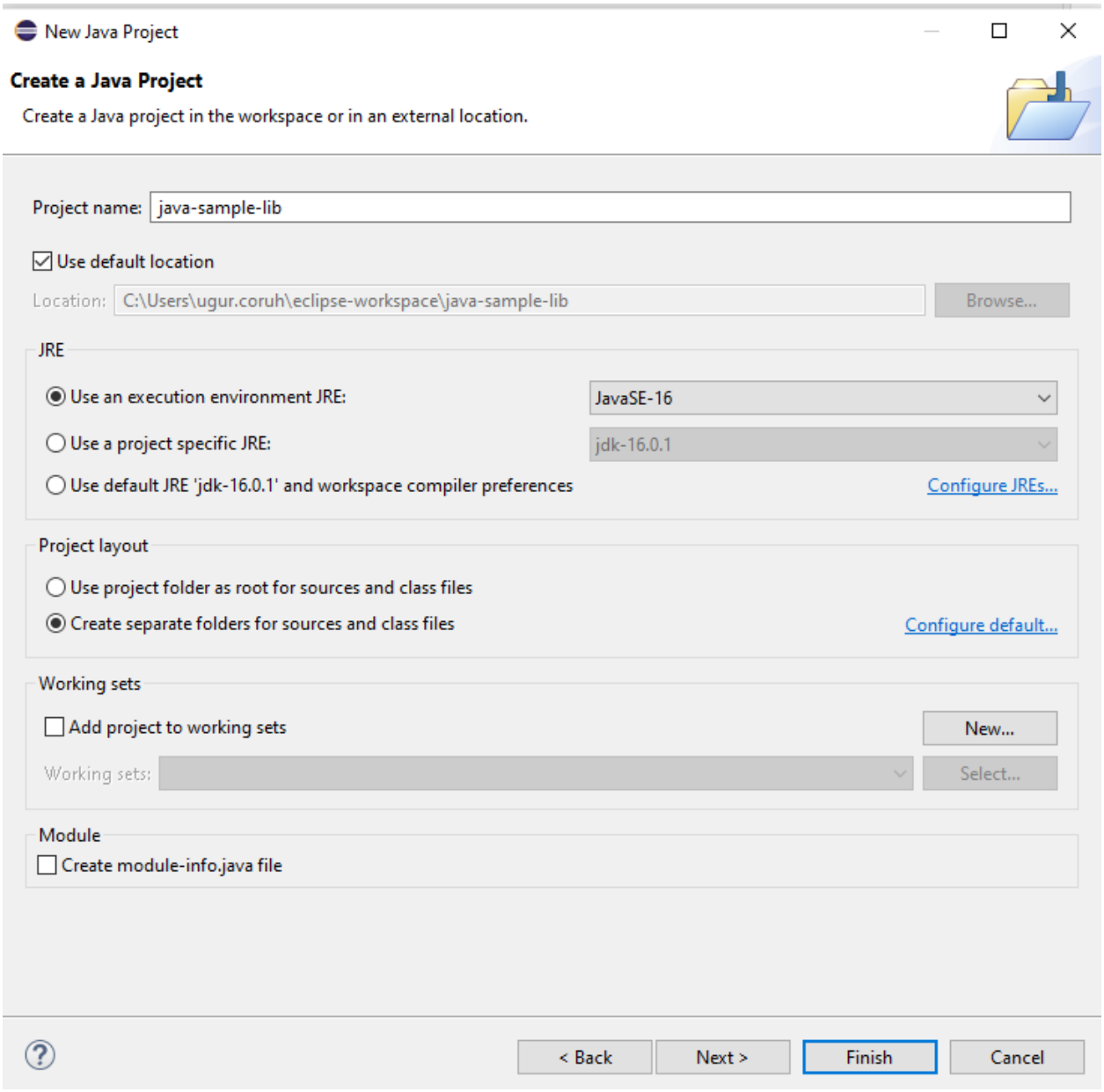


---

select java project

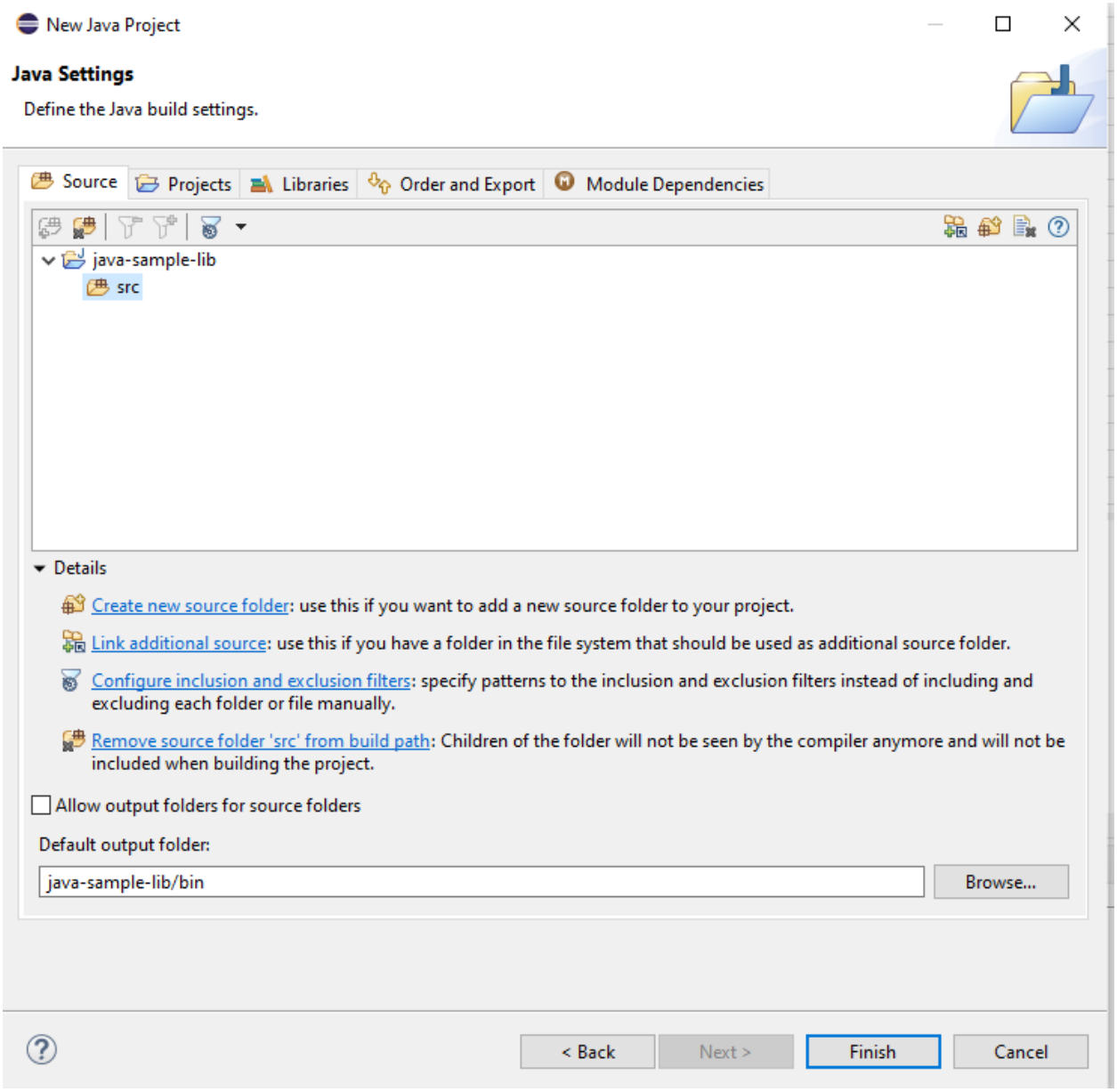


give project name



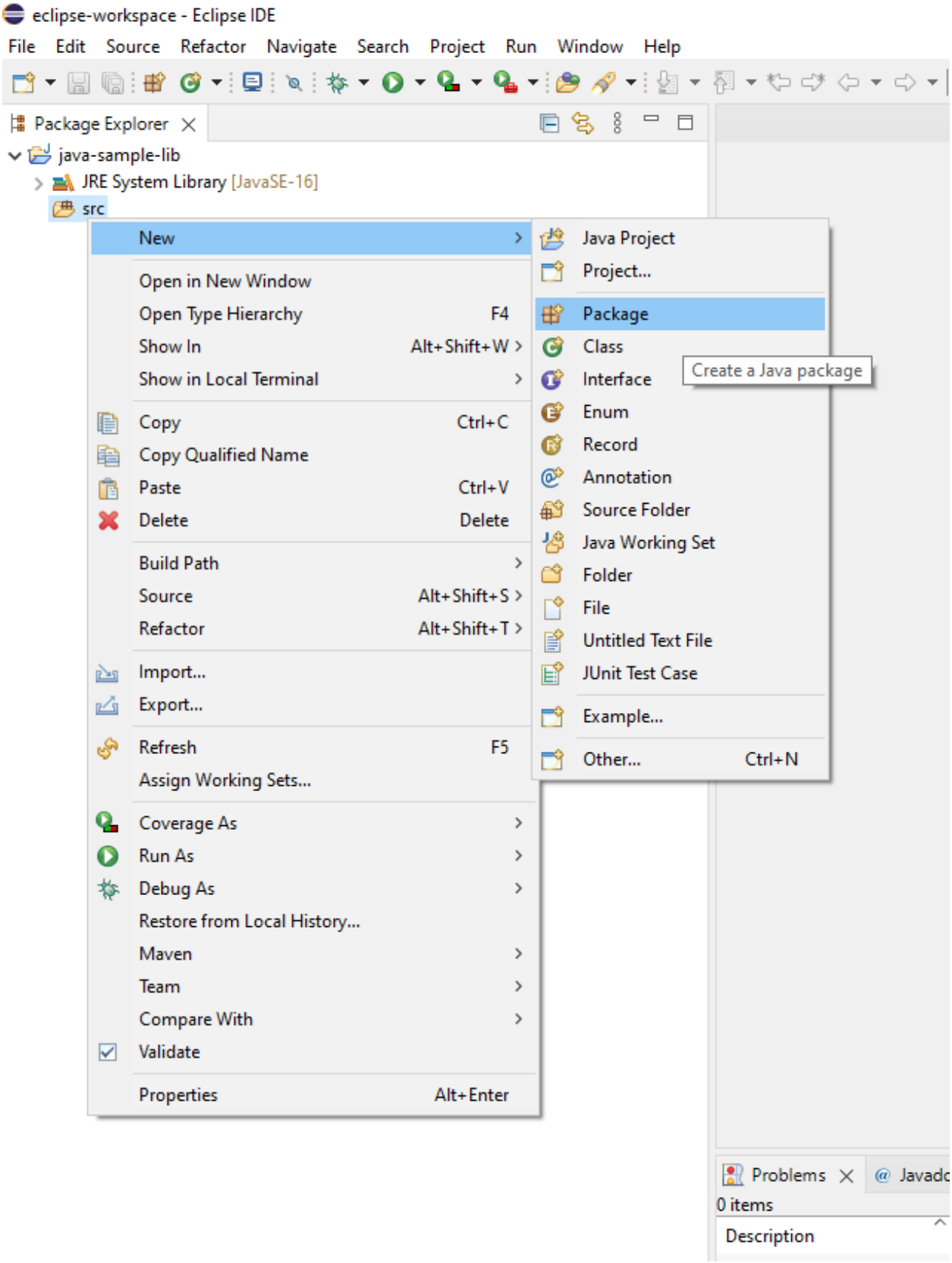
---

select finish



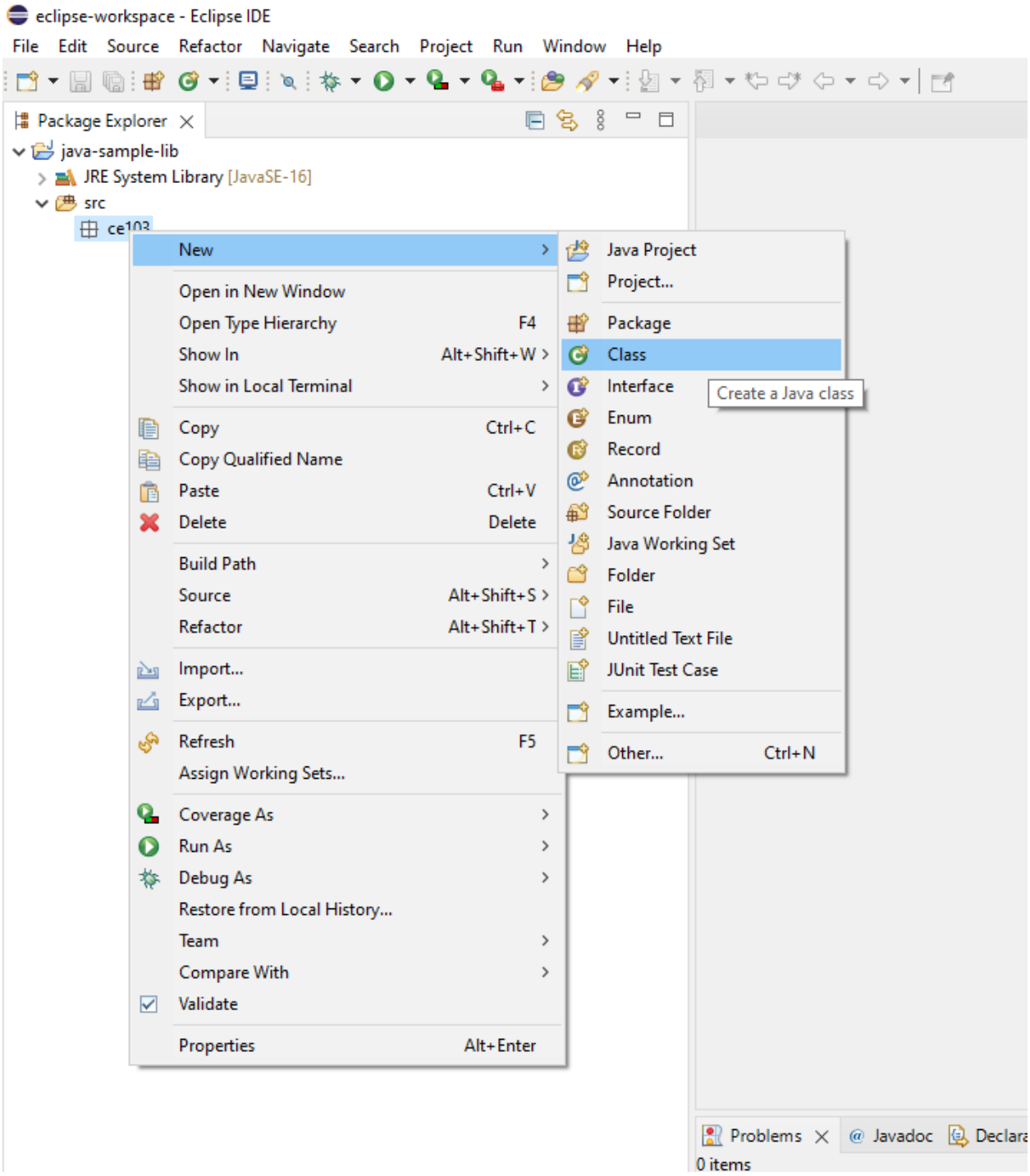
---

first we need to add a default package to keep everything organized



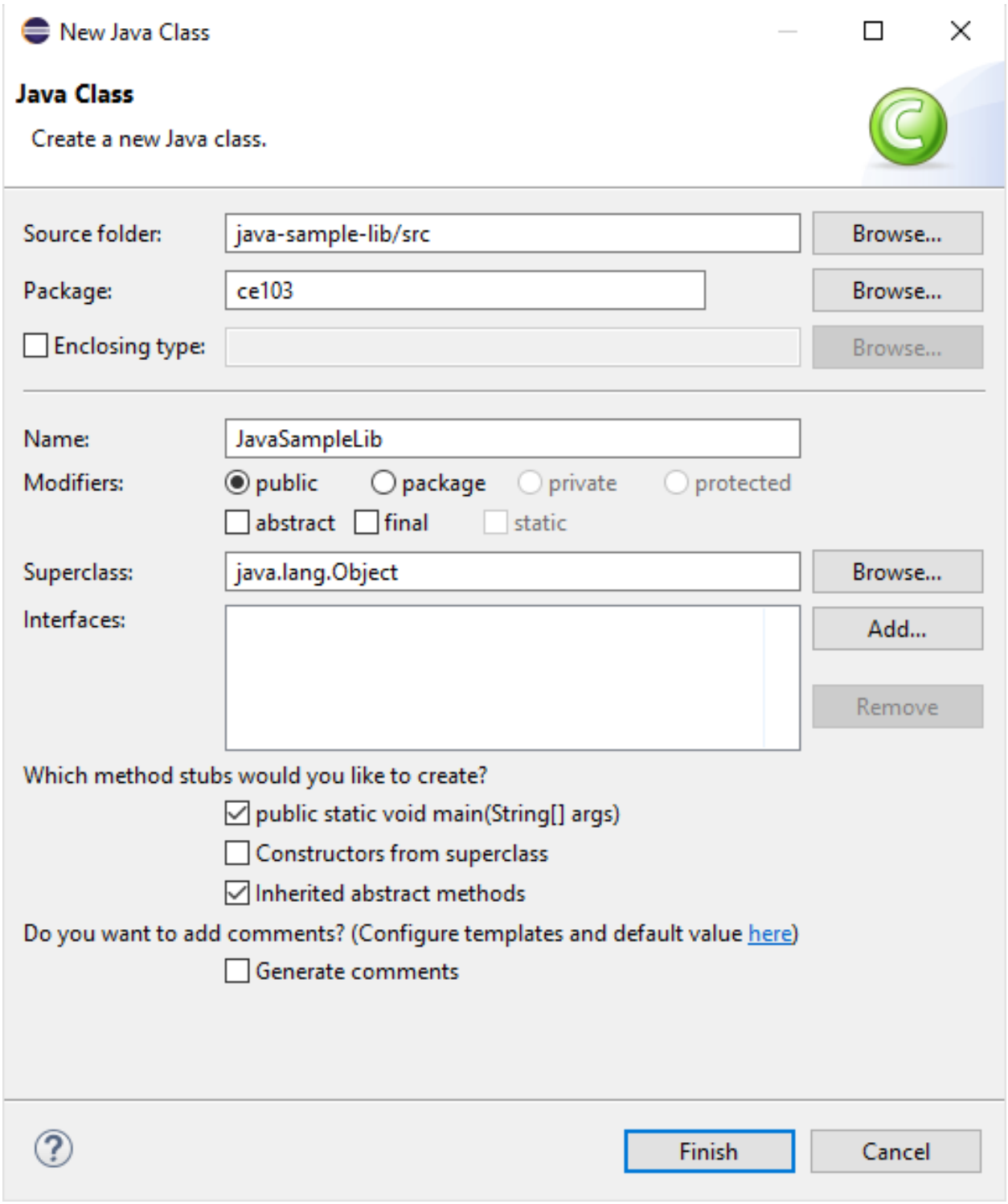
---

then we can create our class that includes our functions



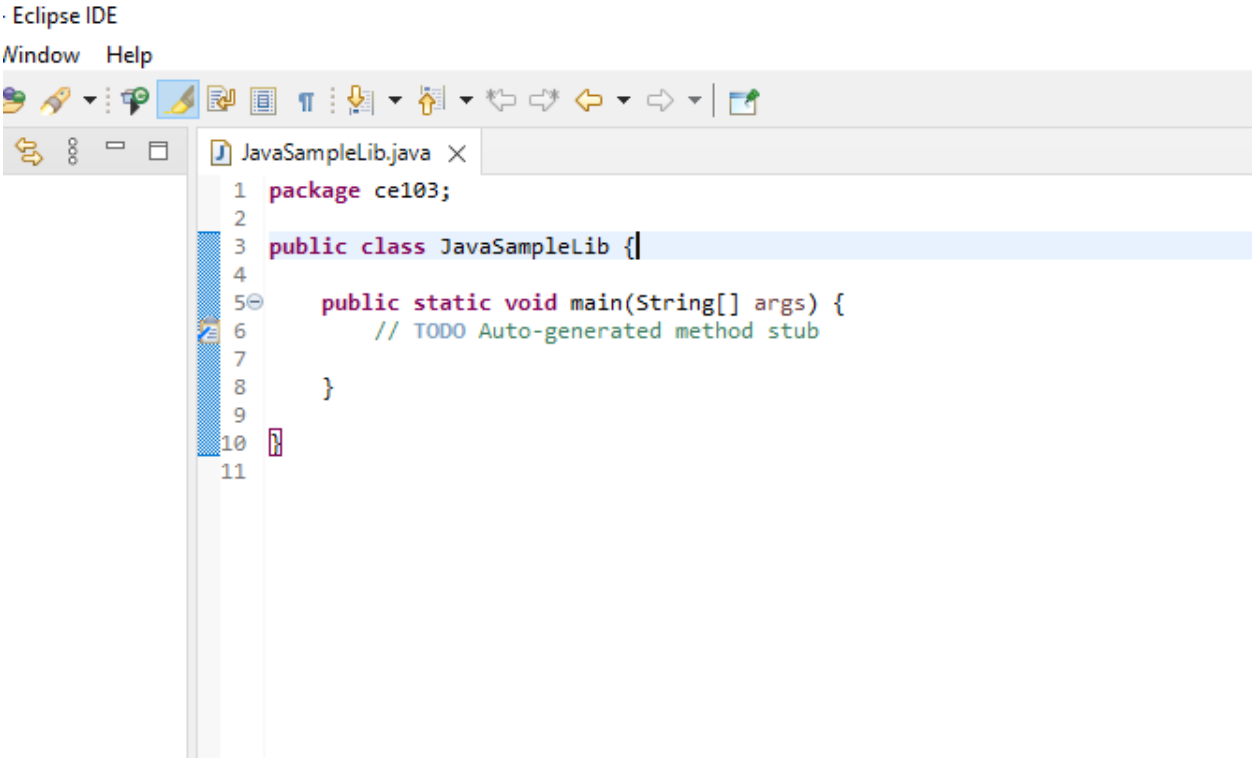
---

give class a name



---

you will have following class with main



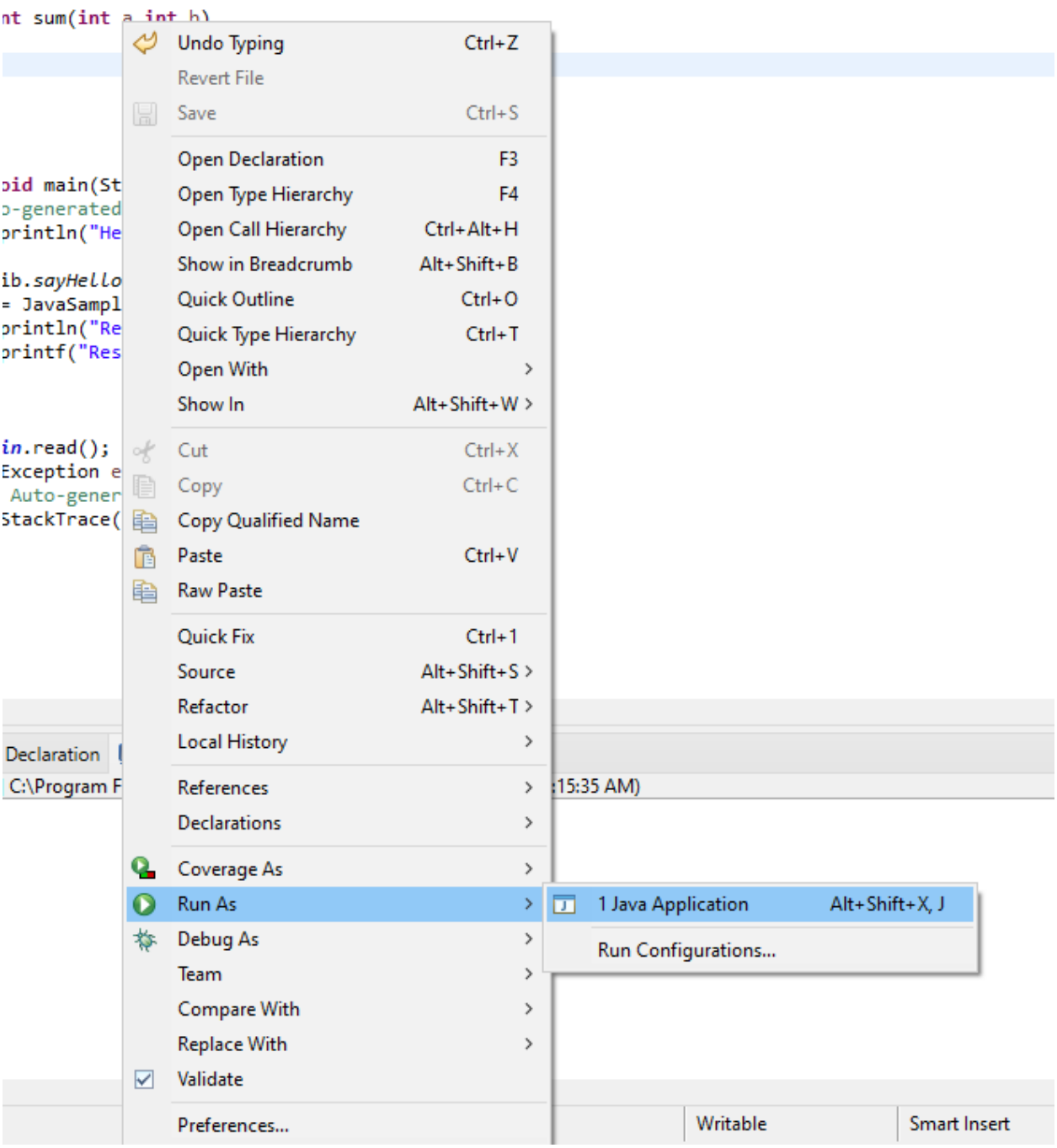
We will create sample java library with static functions as below.

```
package ce103;

import java.io.IOException;

public class JavaSampleLib {

    public static void sayHelloTo(String name) {
        if(name.isBlank() || name.isEmpty())
        {
            System.out.println("Hello "+name);
        }else {
            System.out.println("Hello There");
        }
    }

    public static int sum(int a,int b)
    {
        int c = 0;
        c = a+b;
        return c;
    }

    public static void main(String[] args) {
        // TODO Auto-generated method stub
        System.out.println("Hello World!");

        JavaSampleLib.sayHelloTo("Computer");
        int result = JavaSampleLib.sum(5, 4);
        System.out.println("Results is" + result);
```

```
        System.out.printf("Results is %d \n", result);


        try {
            System.in.read();
        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }

    }

}
```

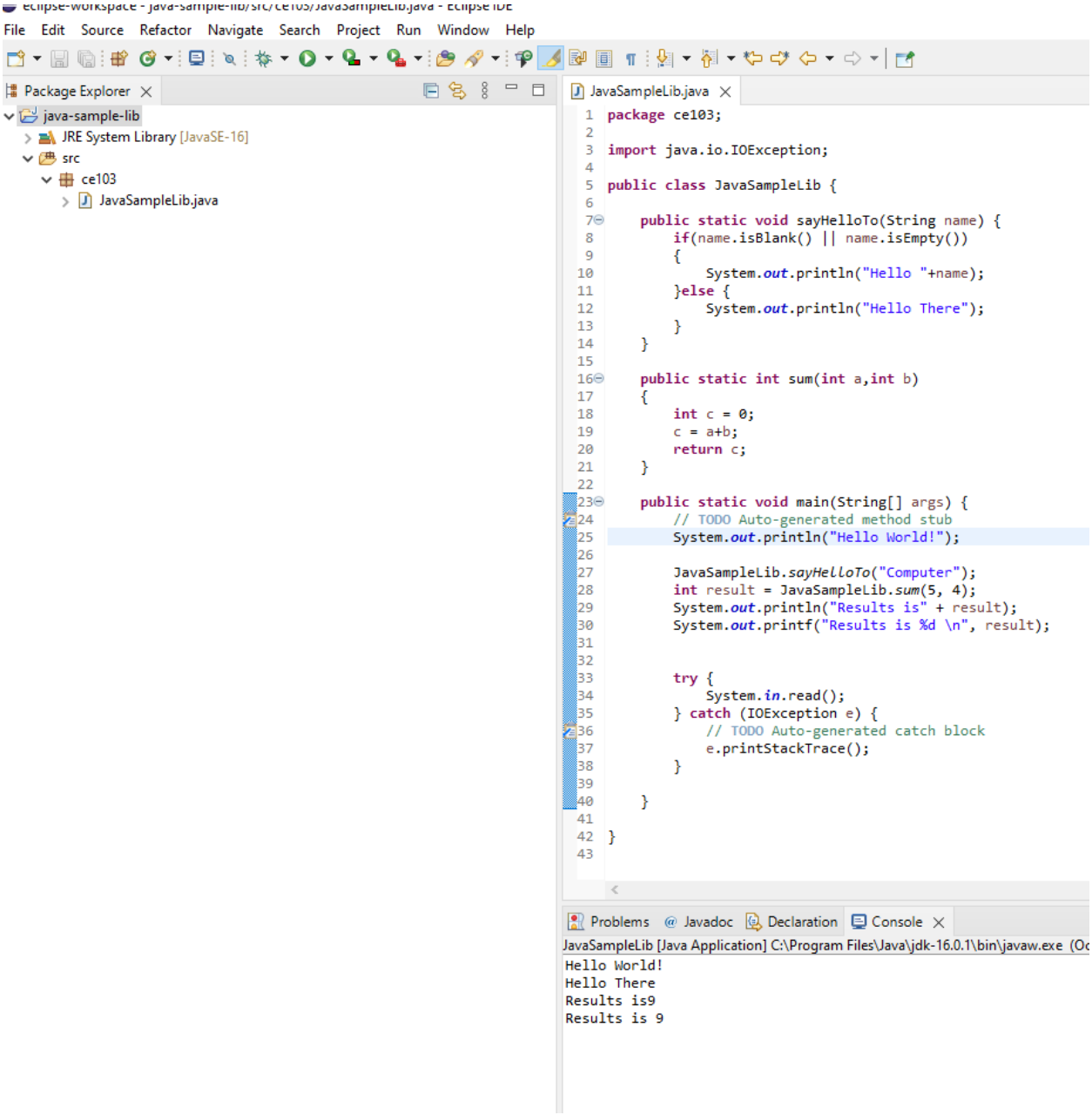
---

also we can add main method to run our library functions. If we run this file its process main function

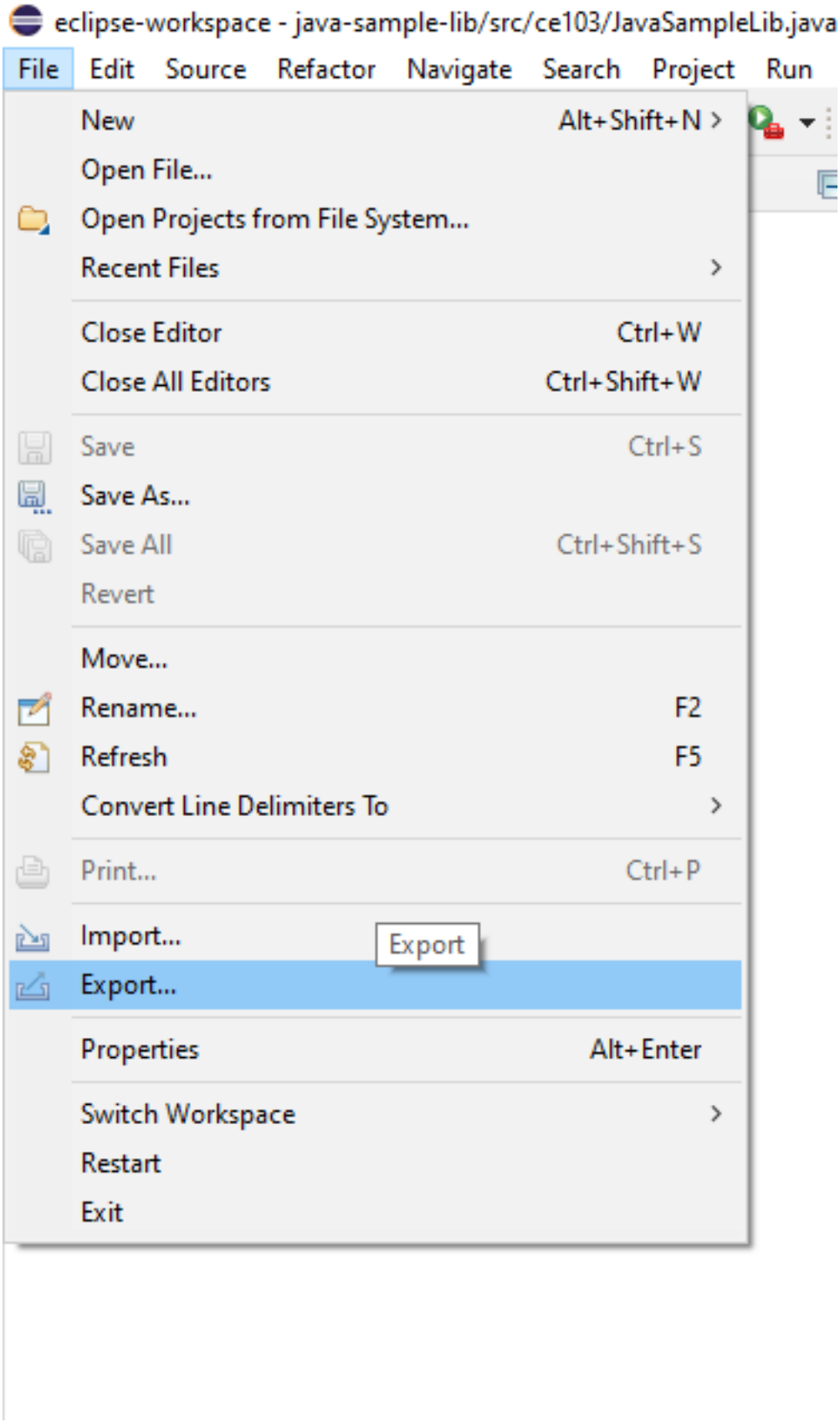


---

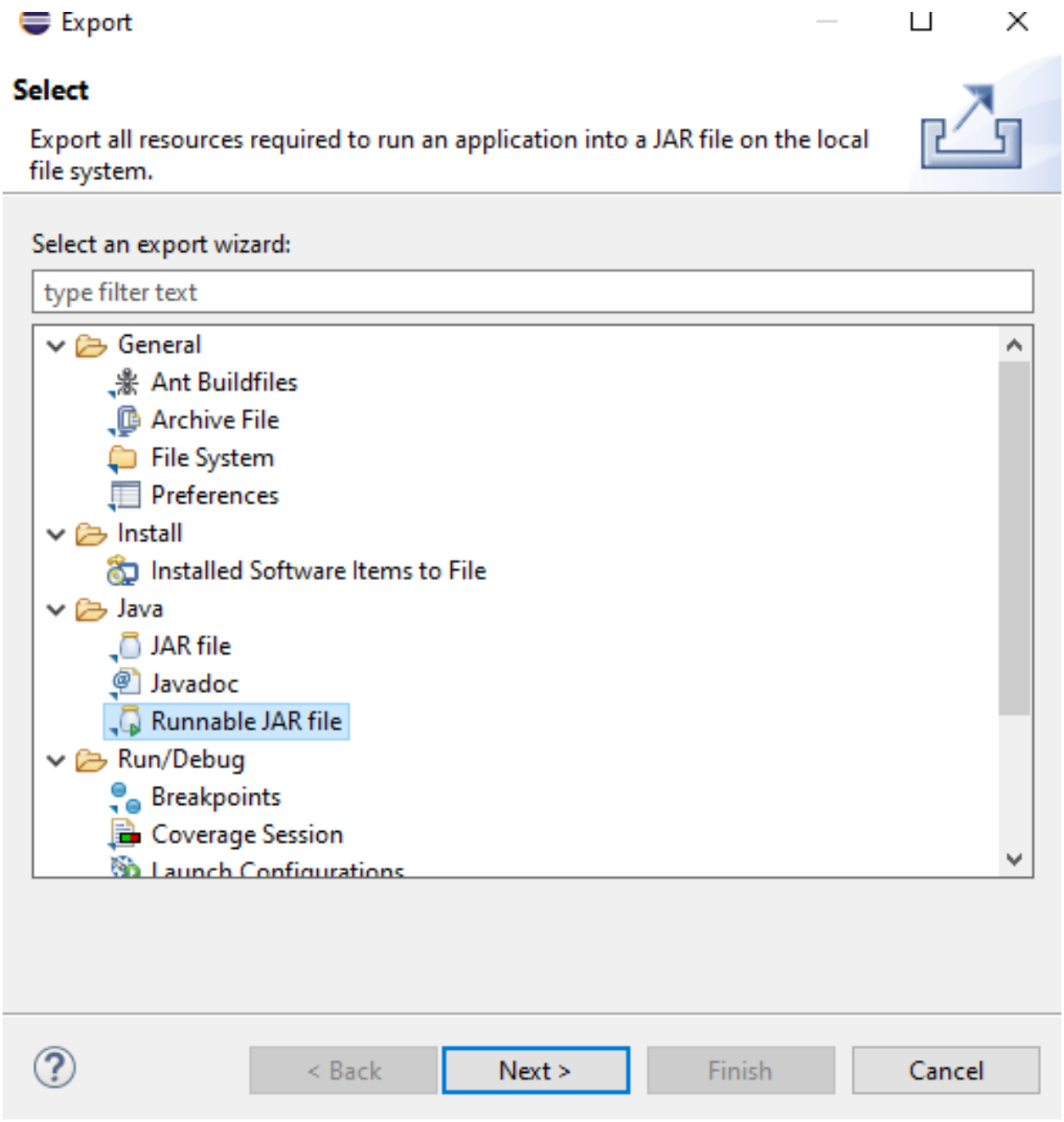
we can see output from console as below



There is no exe files java runtime environment run class files but we can export this as an executable.

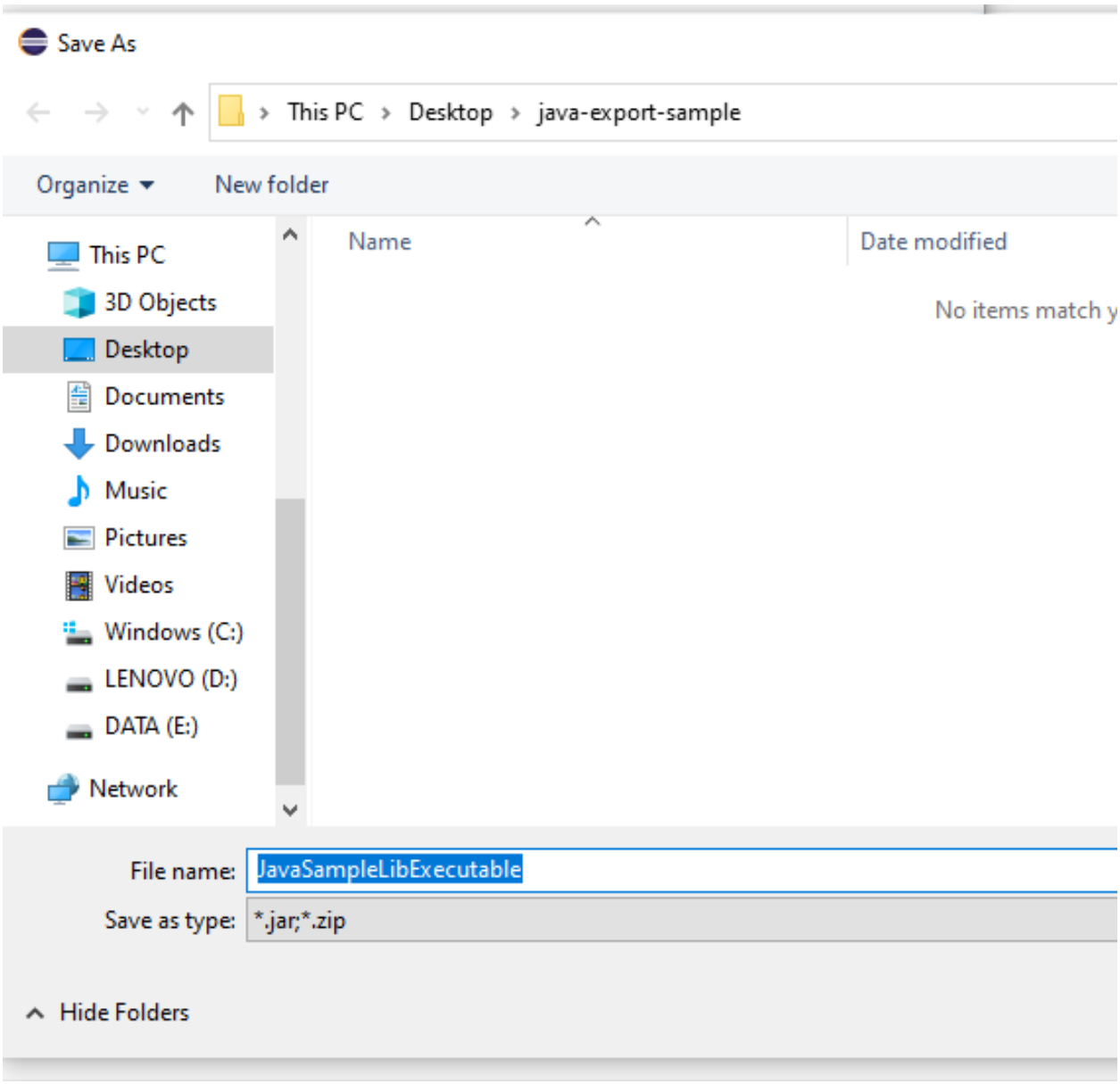


Select Java->Runnable JAR File



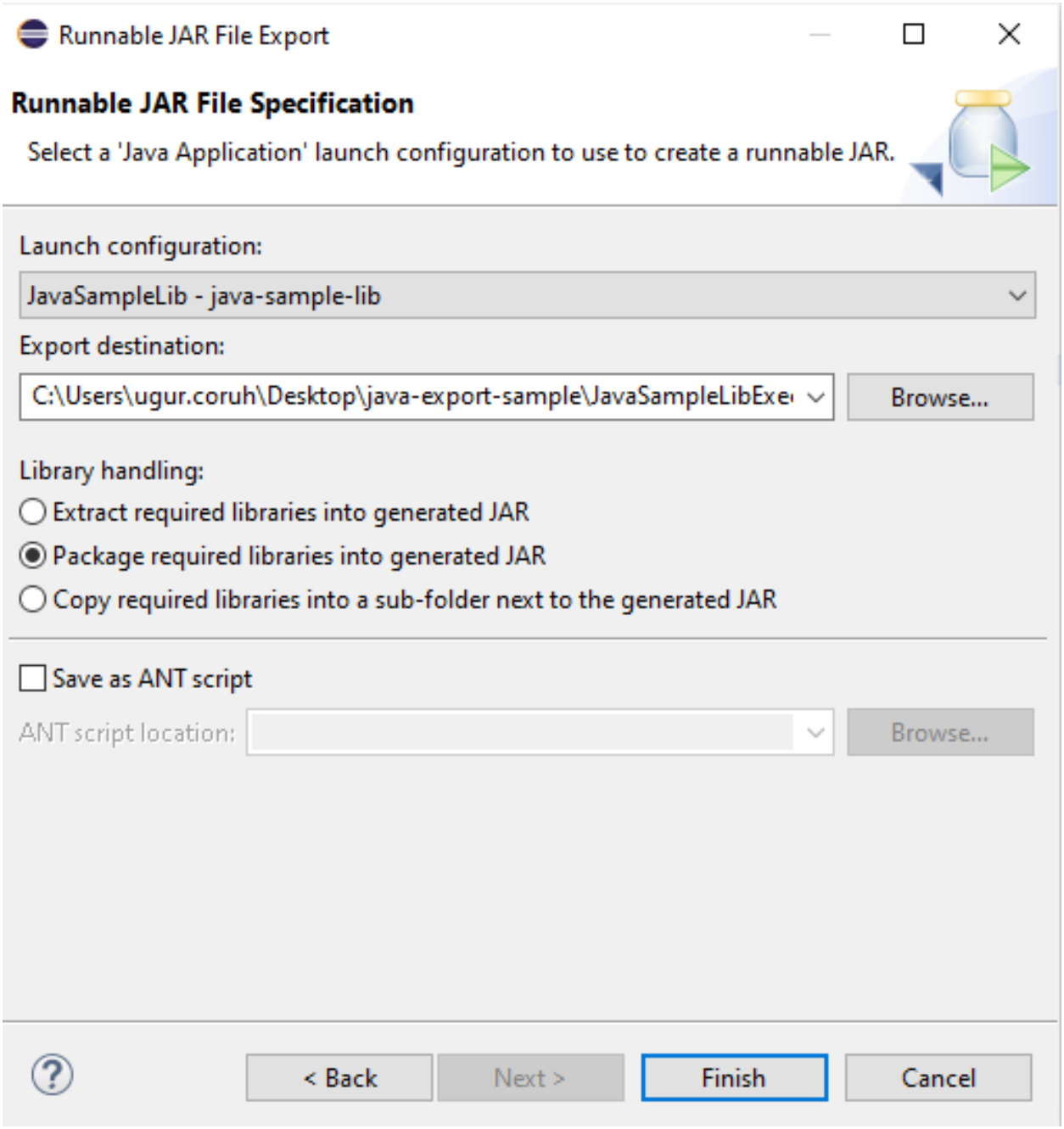
---

click next and set output path for jar file

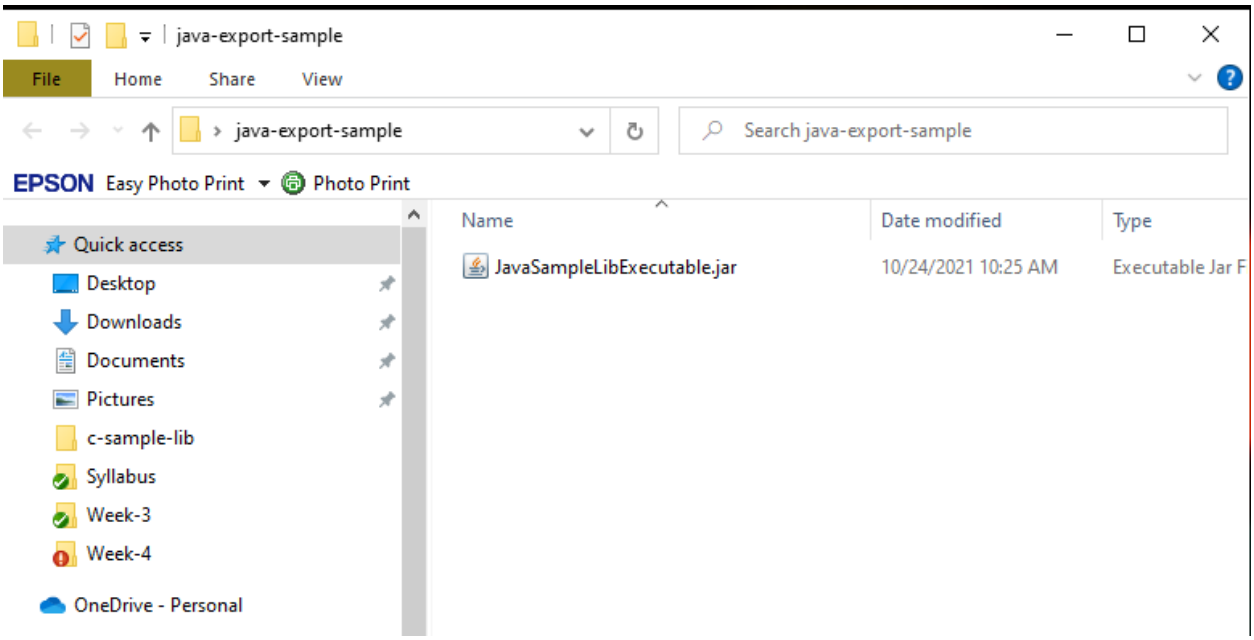


If our project has several external dependecy then we can extract this required files (jar, so, dll) in seperated folder or we can combine them and generate a single executable jar

Lets pack everthing together, Select launch configuration that has main function



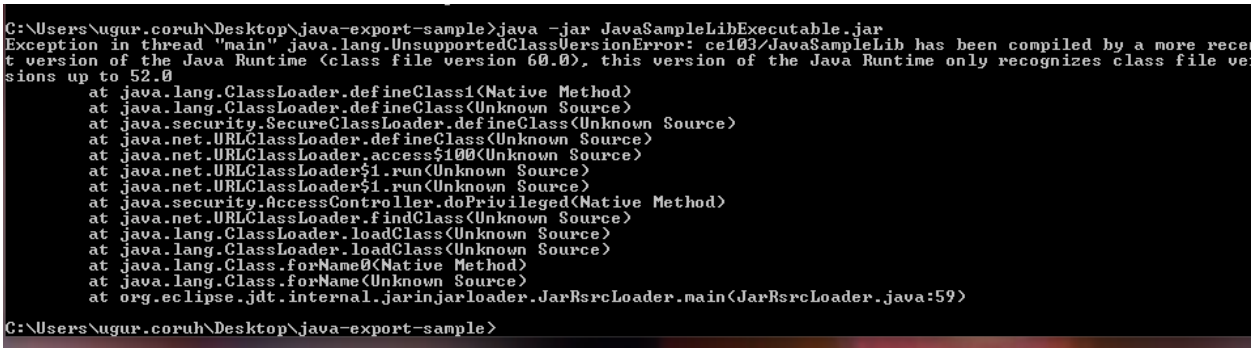
end of this operation we will have the following jar that we can by click



When you click application if cannot run then try command line to see problem

enter jar folder and run the following command

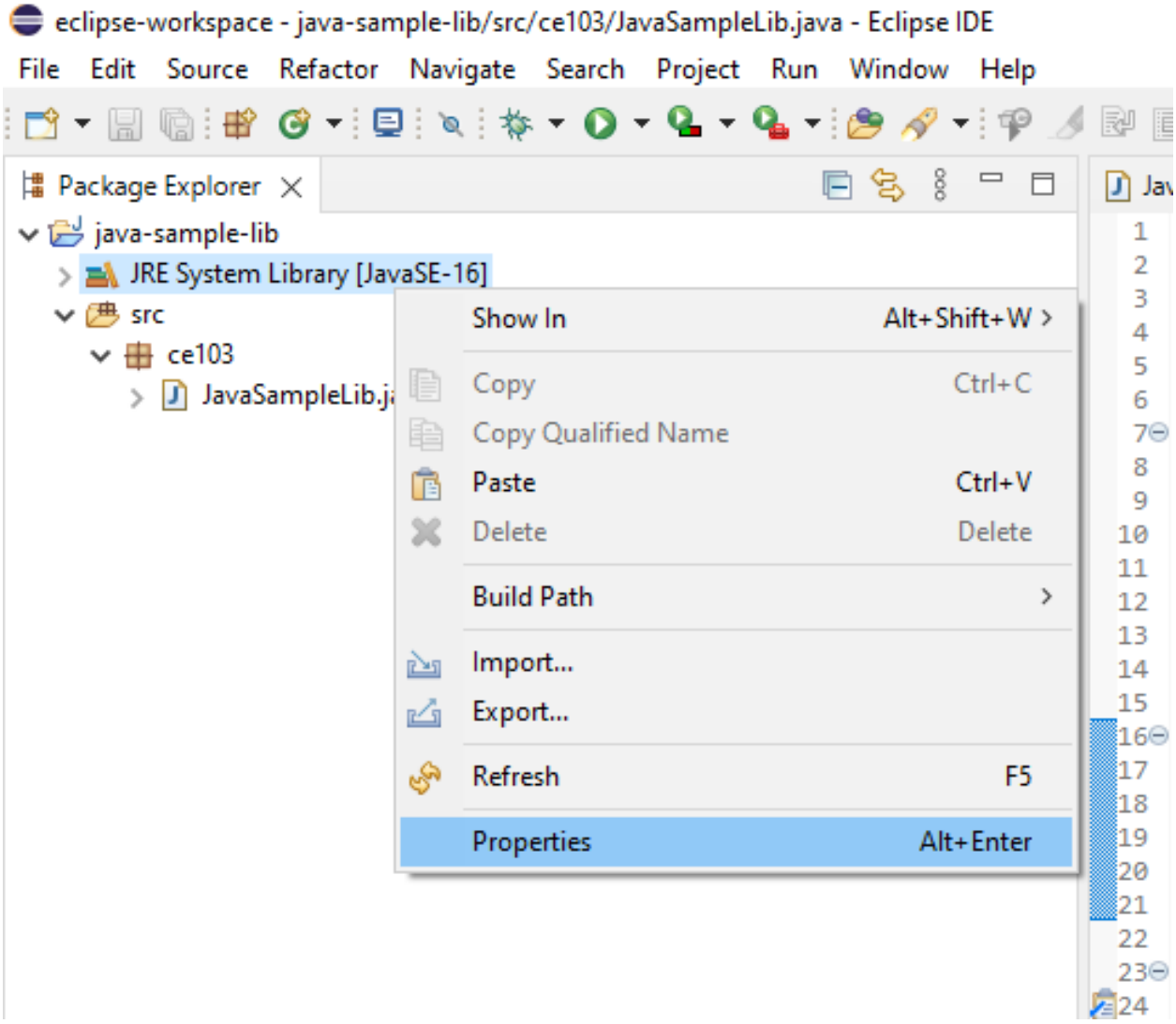
```
java -jar JavaSampleLibExecutable.jar
```



In my case eclipse build JDK is newer than that I installed and set for my OS

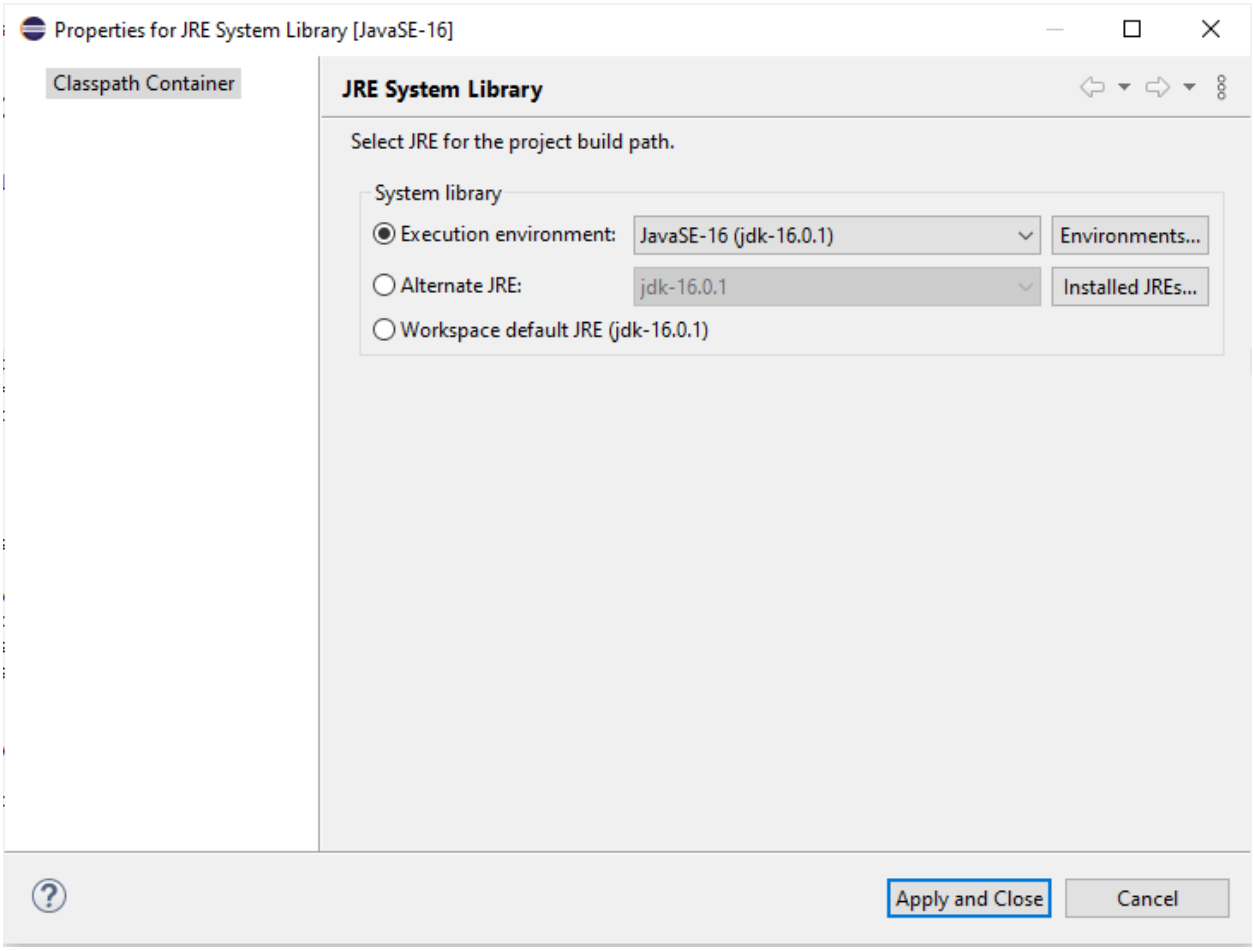
If we check version we can see problem Java version 1.8.0_231

We can found installed and builded JDK for our application from Eclipse setting

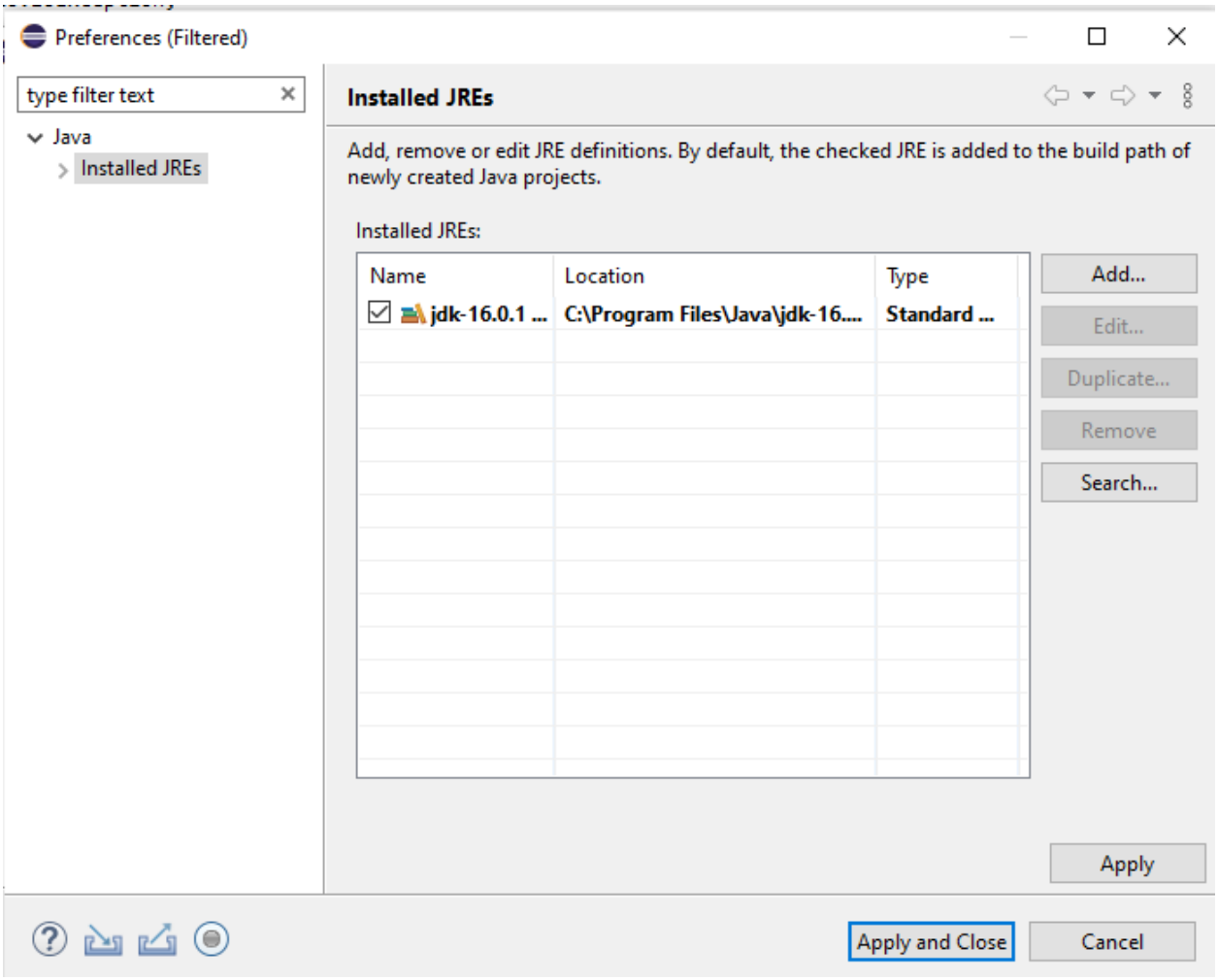


---

select environments



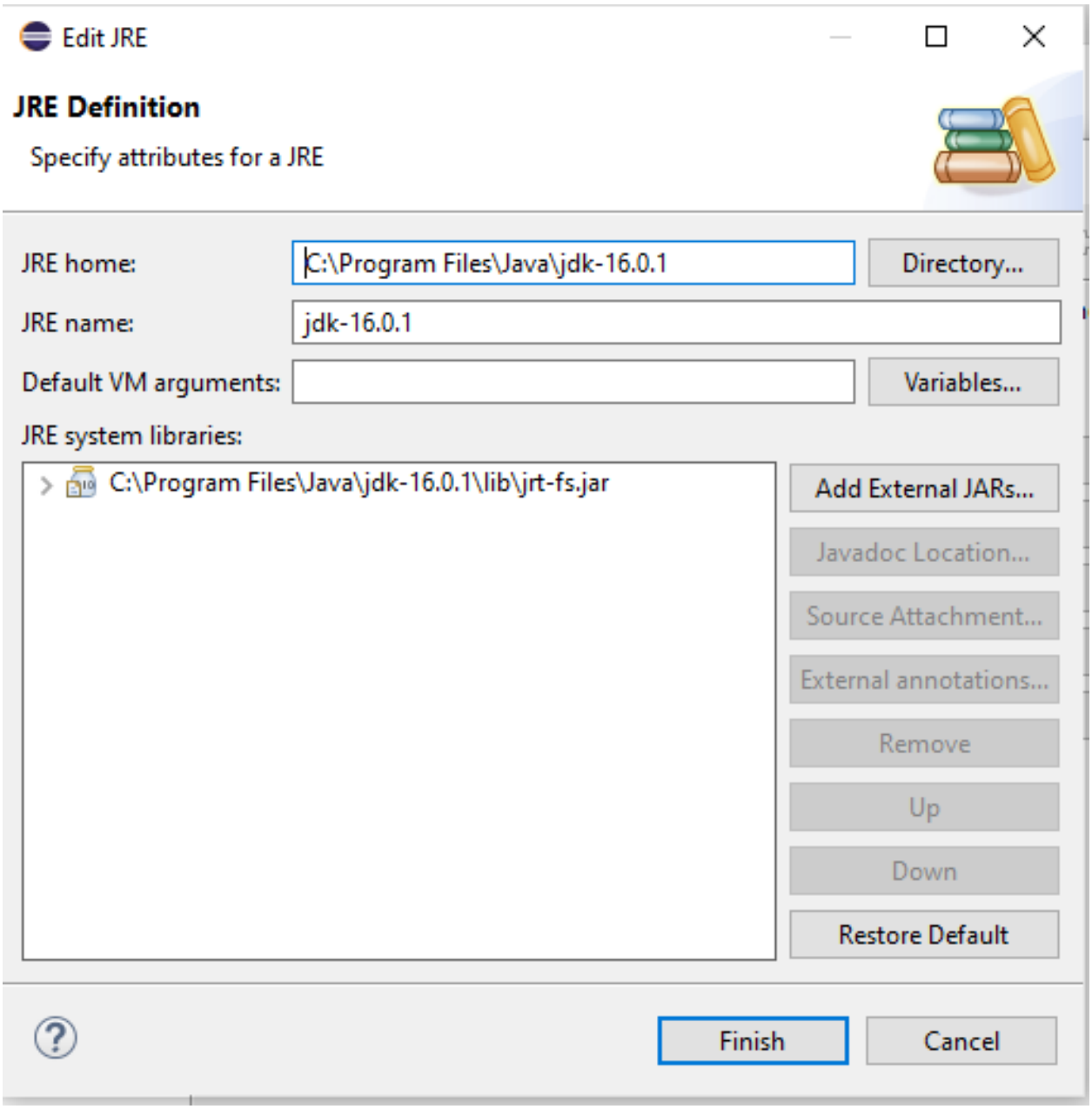
select installed JRE or JDK



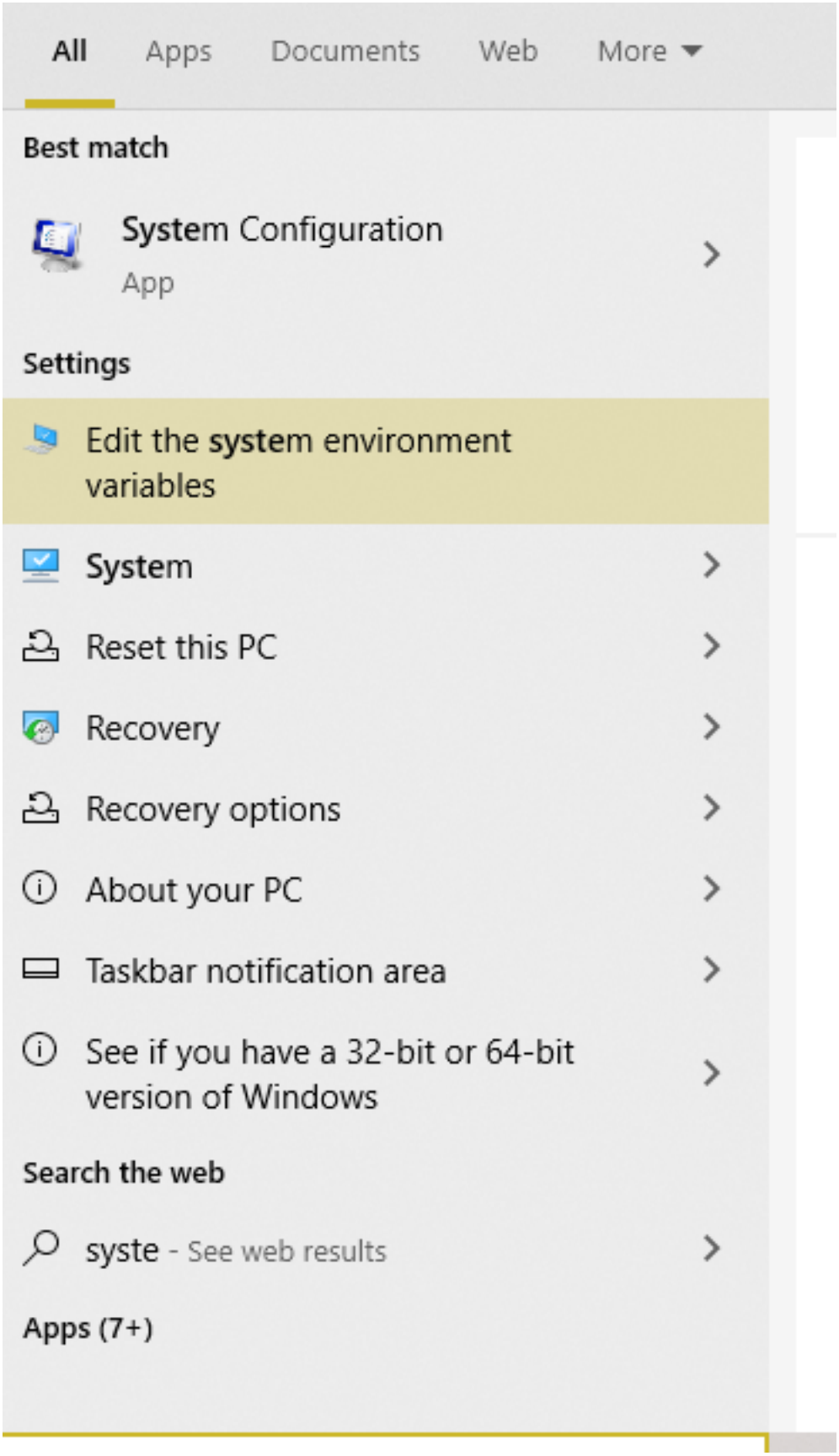
---

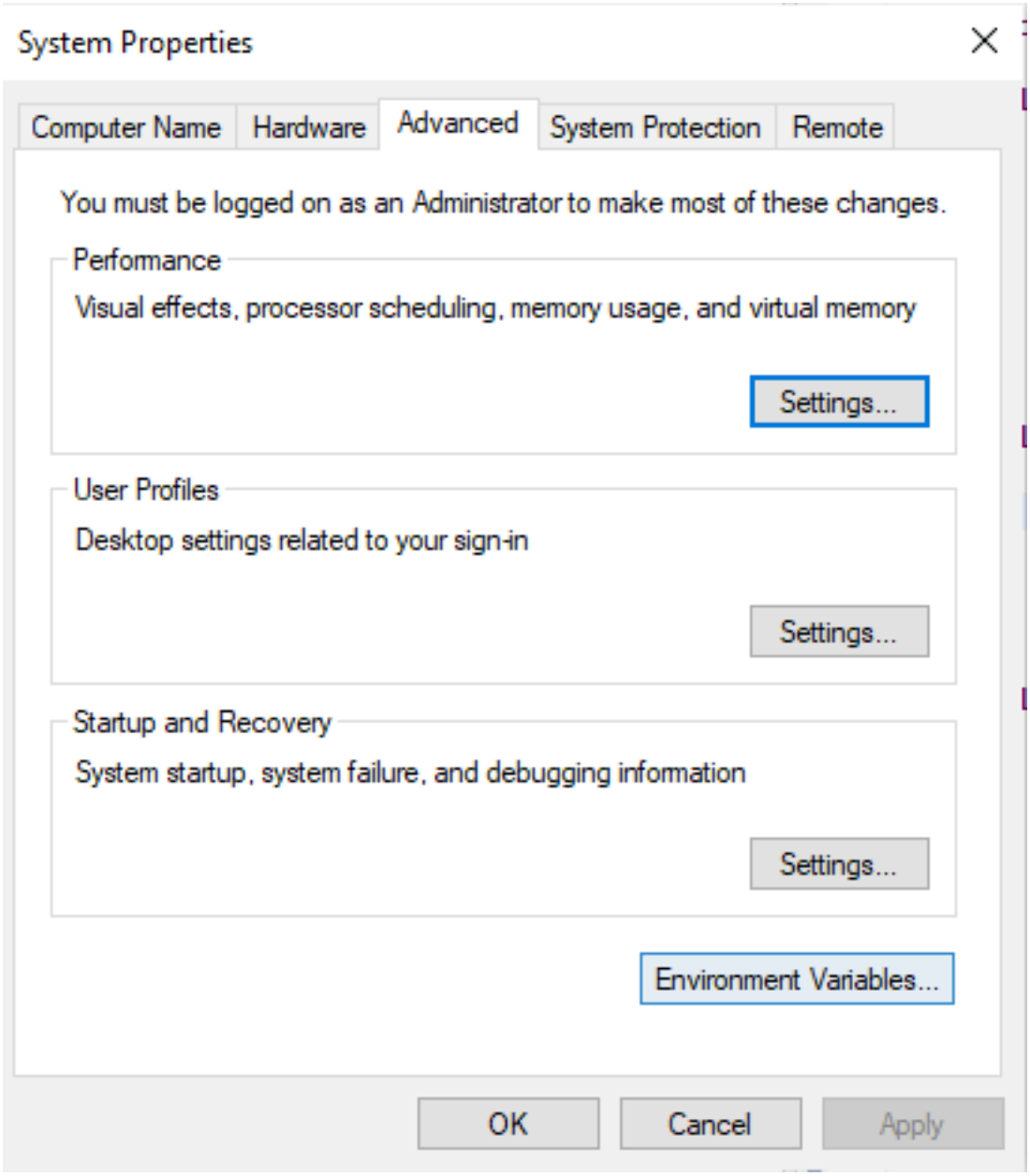
you can see installed JRE or JDK home

`C:\Program Files\Java\jdk-16.0.1`

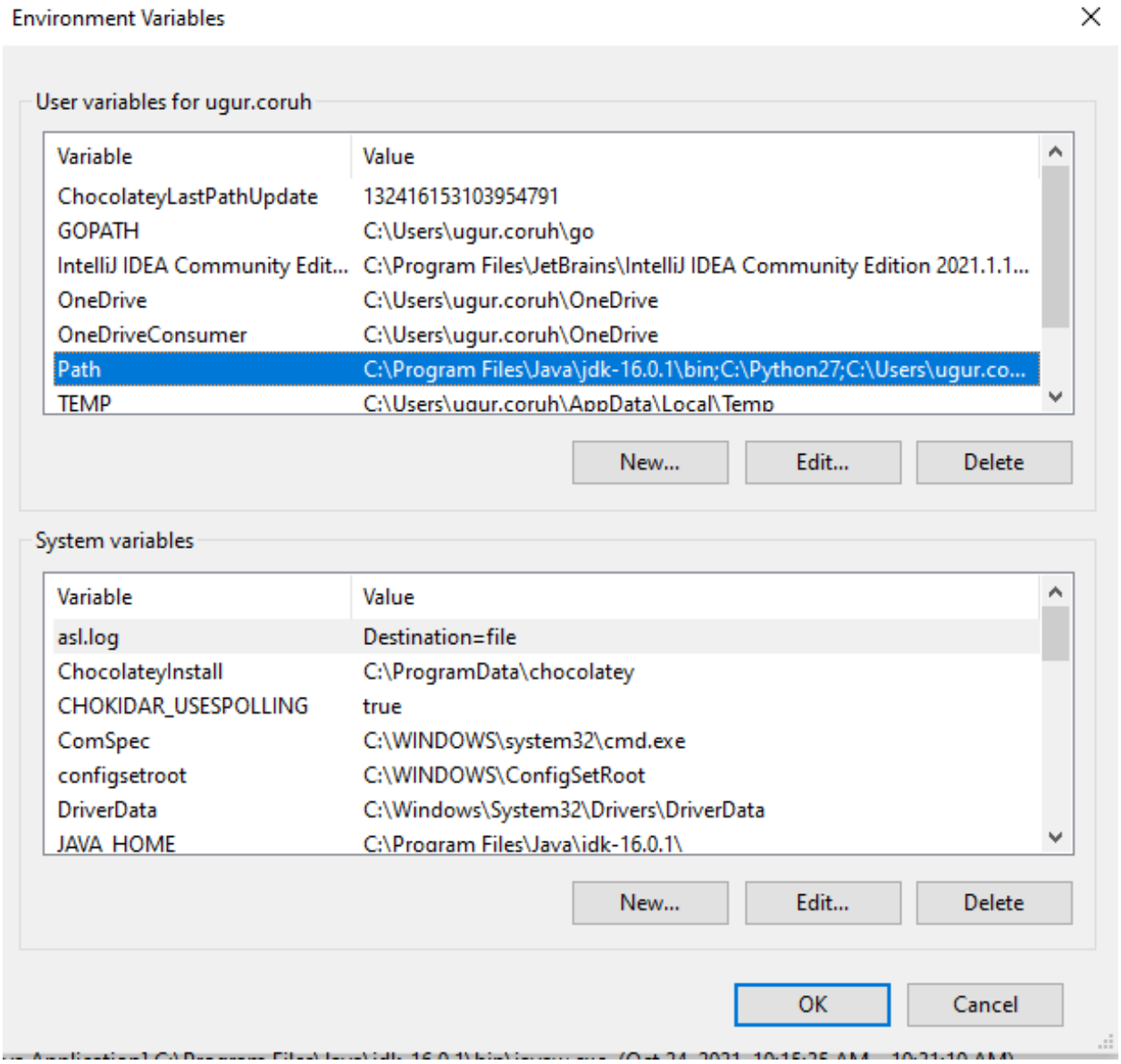


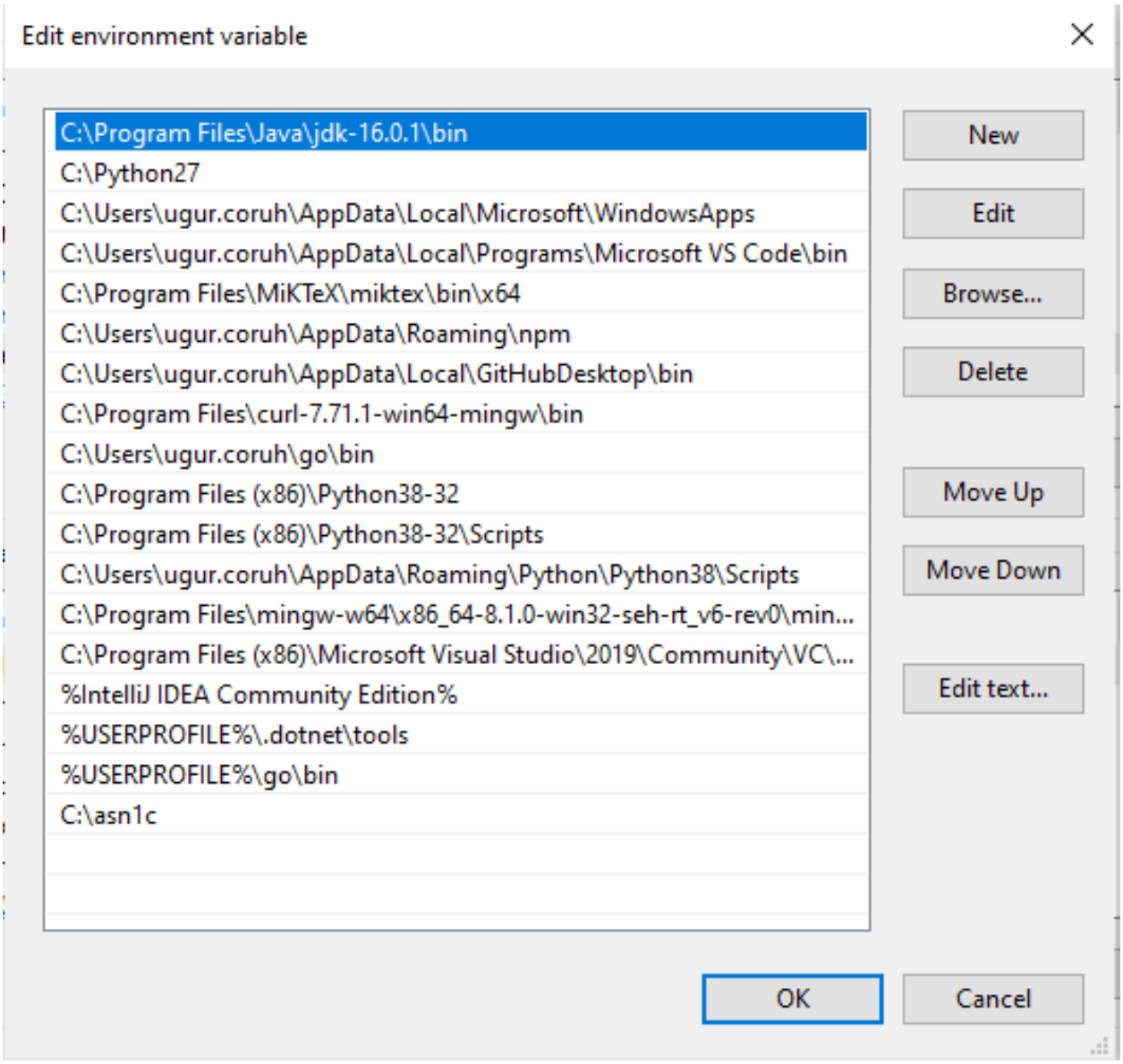
Open system environment to fix this problem


All
Apps
Documents
Web
More
Best match
System Configuration
App
Settings
Edit the system environment variables
System
Reset this PC
Recovery
Recovery options
About your PC
Taskbar notification area
See if you have a 32-bit or 64-bit version of Windows
Search the web
syste - See web results
Apps (7+)

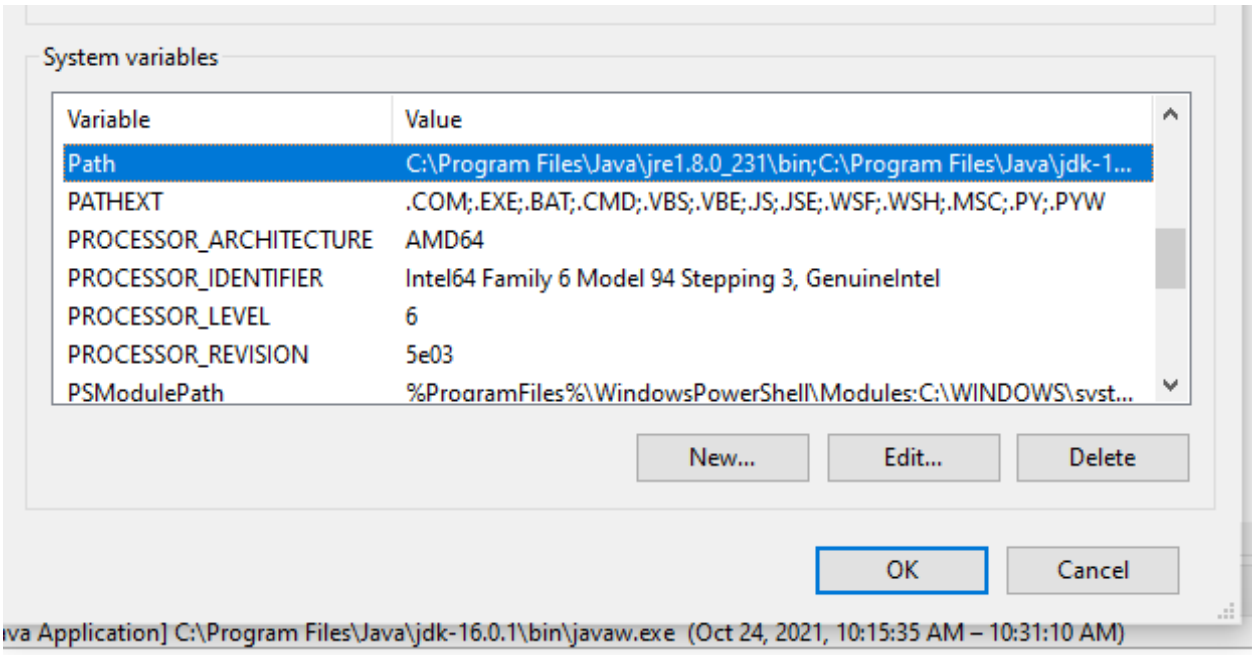


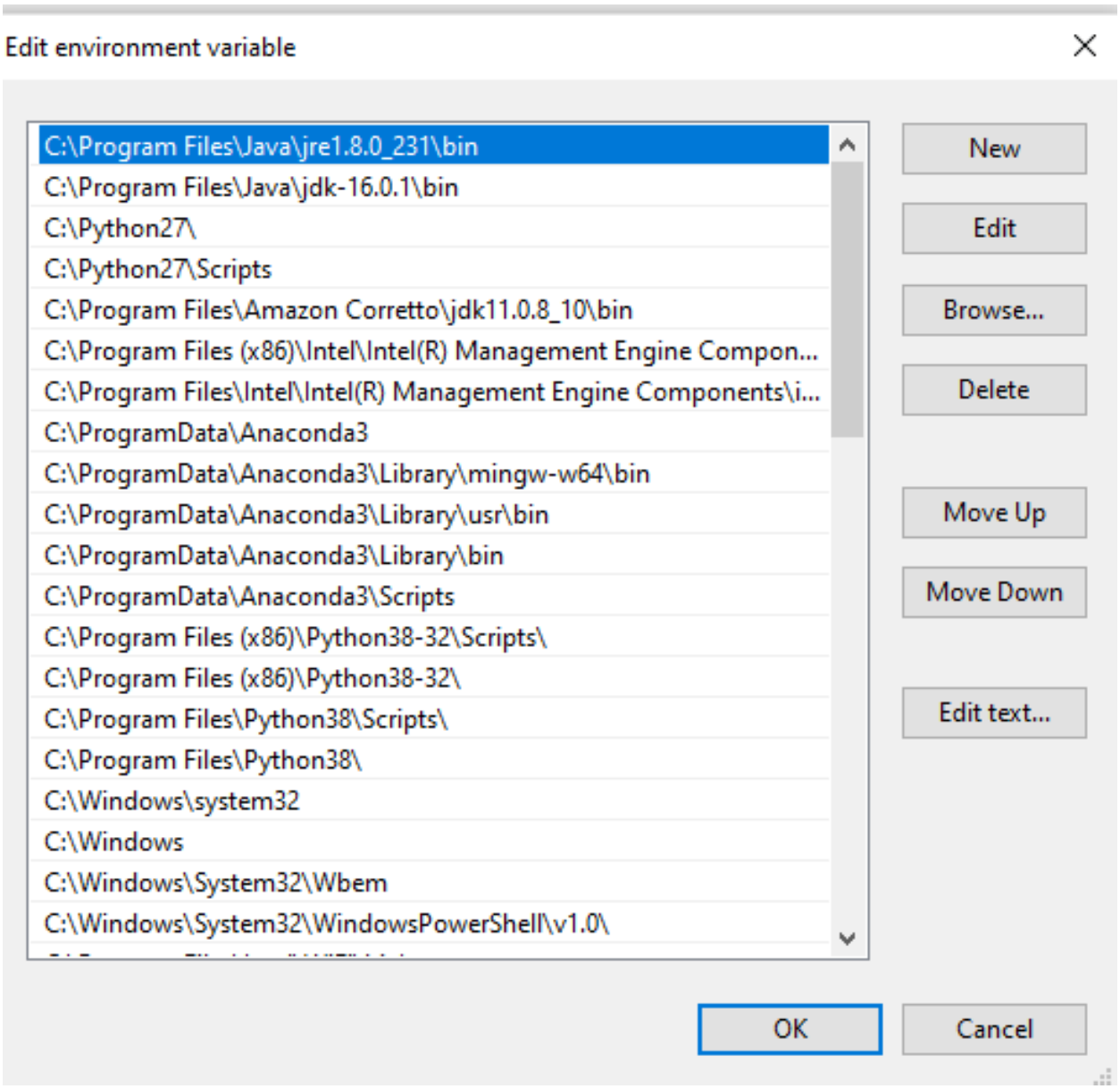
Check user settings first


Environment Variables
User variables for ugur.coruh
Variable
Value
ChocolateyLastPathUpdate
132416153103954791
GOPATH
C:\Users\ugur.coruh\go
IntelliJ IDEA Community Edit...
C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2021.1.1...
OneDrive
C:\Users\ugur.coruh\OneDrive
OneDriveConsumer
C:\Users\ugur.coruh\OneDrive
Path
C:\Program Files\Java\jdk-16.0.1\bin;C:\Python27;C:\Users\ugur.co...
TEMP
C:\Users\ugur.coruh\AppData\Local\Temp
New...
Edit...
Delete
System variables
Variable
Value
asl.log
Destination=file
ChocolateyInstall
C:\ProgramData\chocolatey
CHOKIDAR_USESPOLLING
true
ComSpec
C:\WINDOWS\system32\cmd.exe
configsetroot
C:\WINDOWS\ConfigSetRoot
DriverData
C:\Windows\System32\Drivers\DriverData
JAVA_HOME
C:\Program Files\Java\jdk-16.0.1\
New...
Edit...
Delete
OK
Cancel



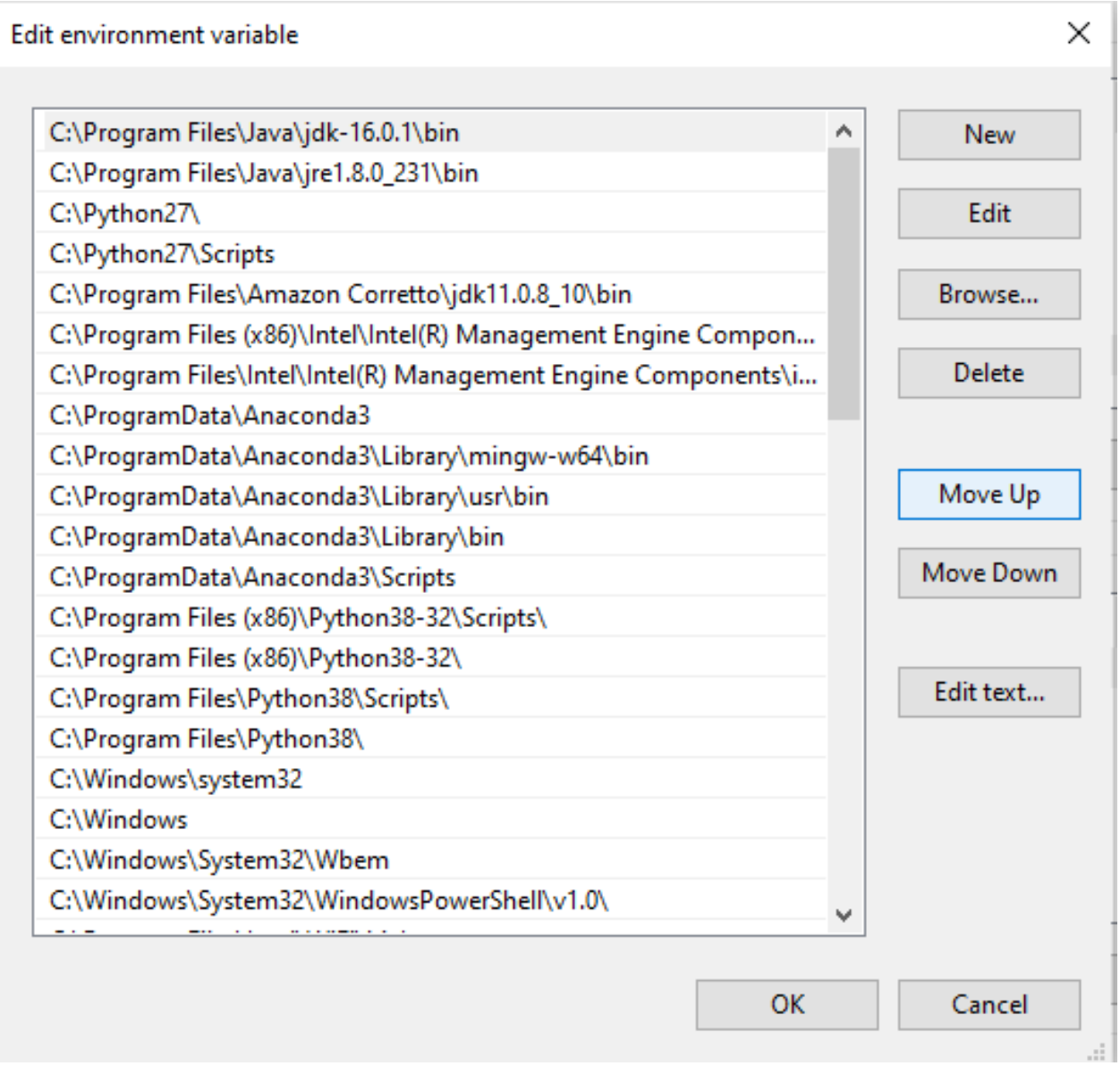
Check system settings


System variables
Variable
Value
Path
C:\Program Files\Java\jre1.8.0_231\bin;C:\Program Files\Java\jdk-1...
PATHEXT
.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC;.PY;.PYW
PROCESSOR_ARCHITECTURE
AMD64
PROCESSOR_IDENTIFIER
Intel64 Family 6 Model 94 Stepping 3, GenuineIntel
PROCESSOR_LEVEL
6
PROCESSOR_REVISION
5e03
PSModulePath
%ProgramFiles%\WindowsPowerShell\Modules;C:\WINDOWS\syst...
New...
Edit...
Delete
OK
Cancel
va Application] C:\Program Files\Java\jdk-16.0.1\bin\javaw.exe (Oct 24, 2021, 10:15:35 AM – 10:31:10 AM)

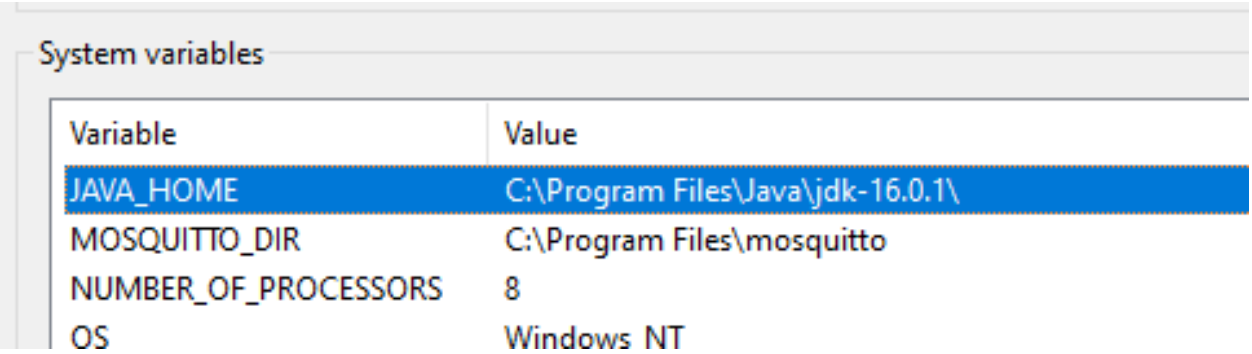


---

we will move up the JDK 16 configuration then command line will run first java



Also in system setting check JAVA_HOME

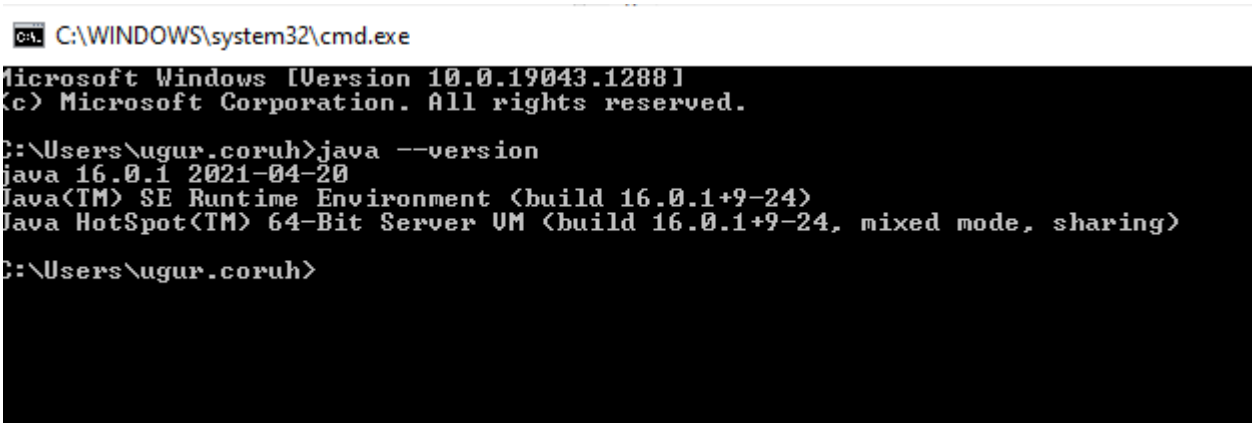


After this settings close current command line and open new one

write

```
java --version
```

if you see java version updated and 16.0.1 then settings are correct



---

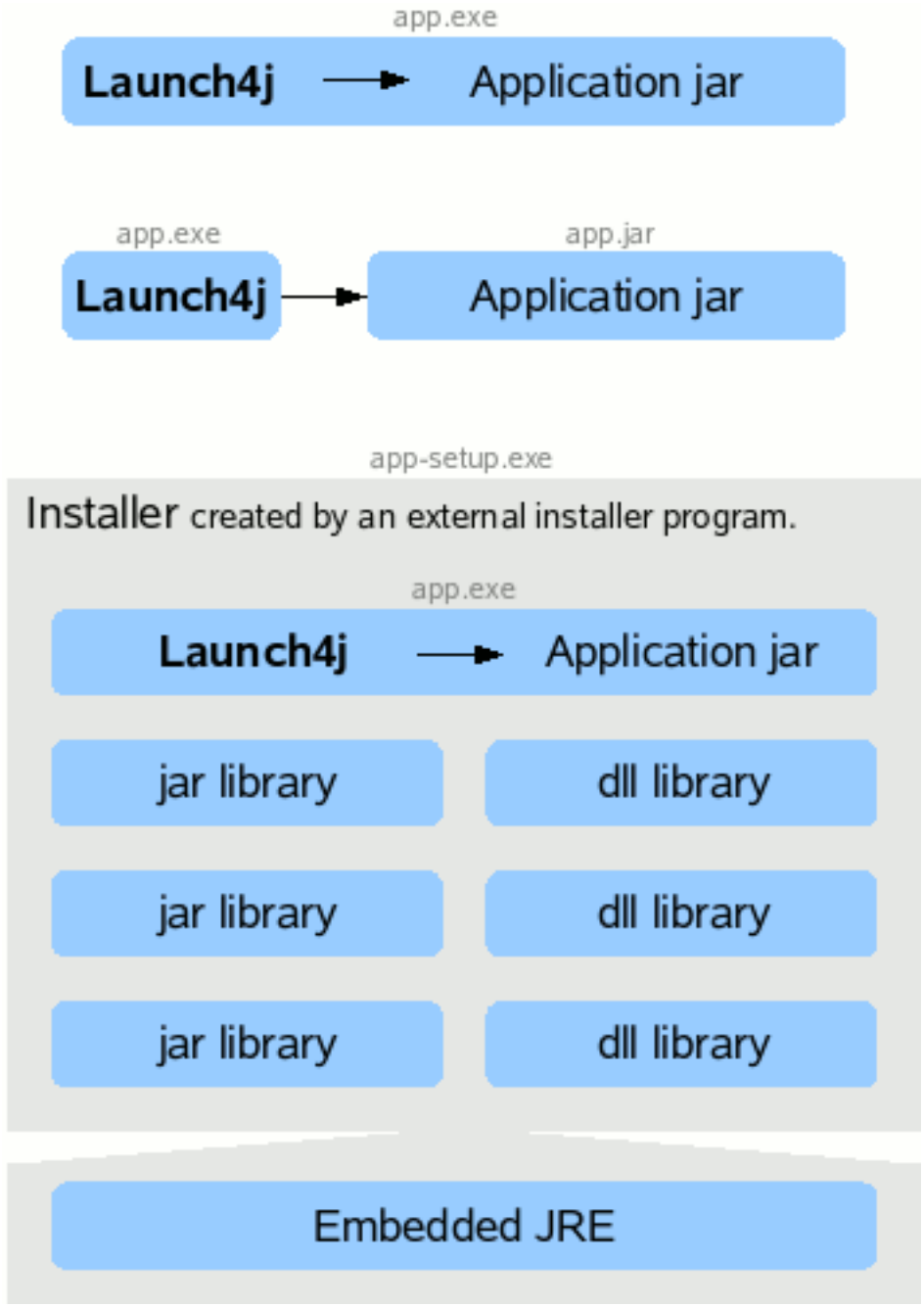
and now if we enter and run application as follow we will see output

But when you click this jar its not running as you see so we have options to provide a clickable application there

Launch4j is an option here

Launch4j - Cross-platform Java executable wrapper[10]

[10] http://launch4j.sourceforge.net/index.html



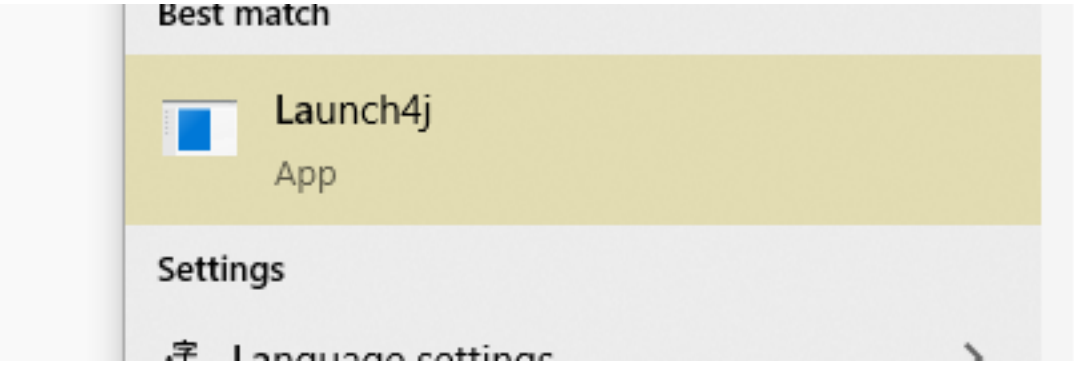
---

you can watch this tutorial also

How to convert jar to exe using Launch4J Full explanation - YouTube[11]

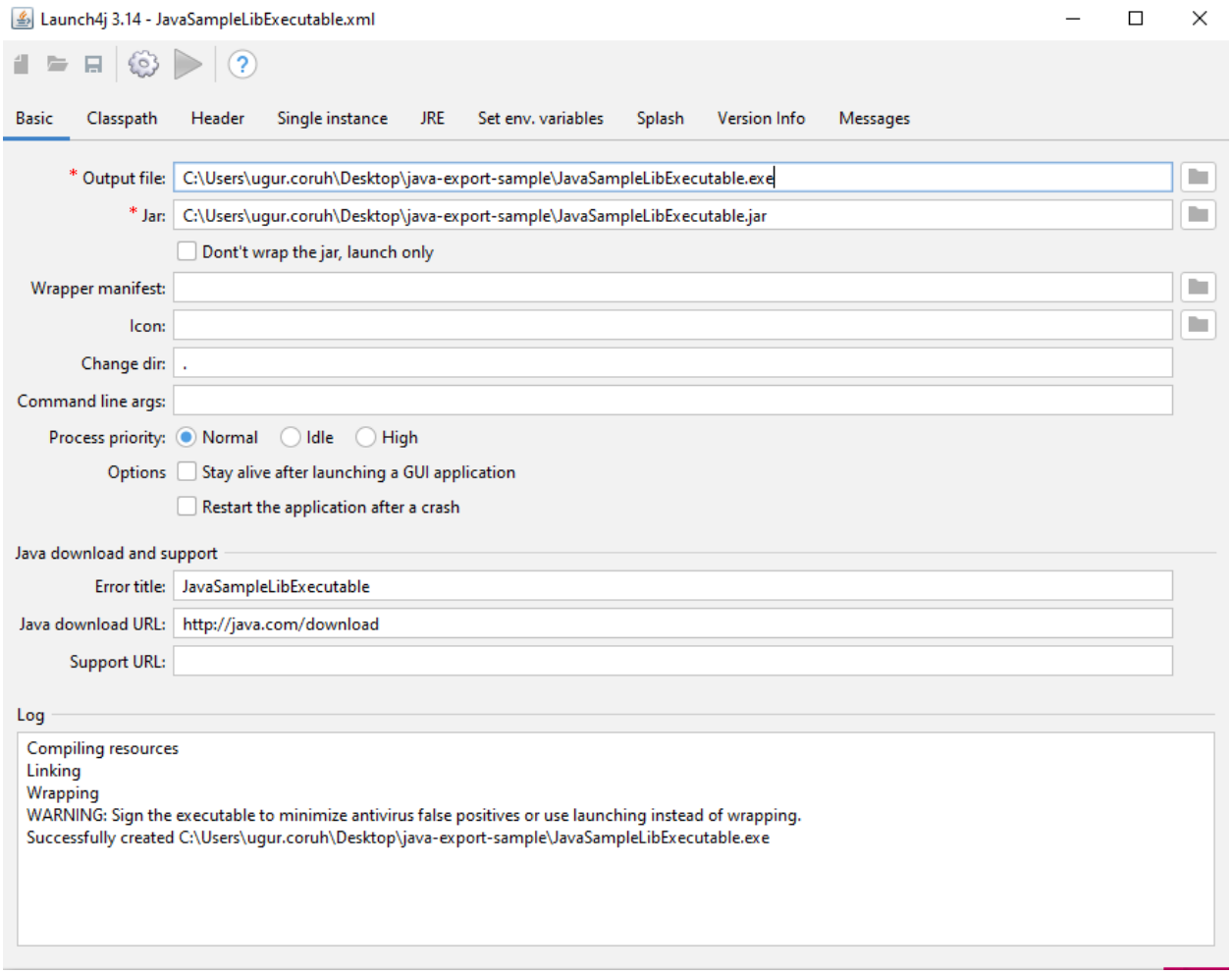
Download and install launch4j and open application

[11] https://www.youtube.com/watch?v=MyMPPuYGN-U&ab_channel=GoXR3PlusStudio



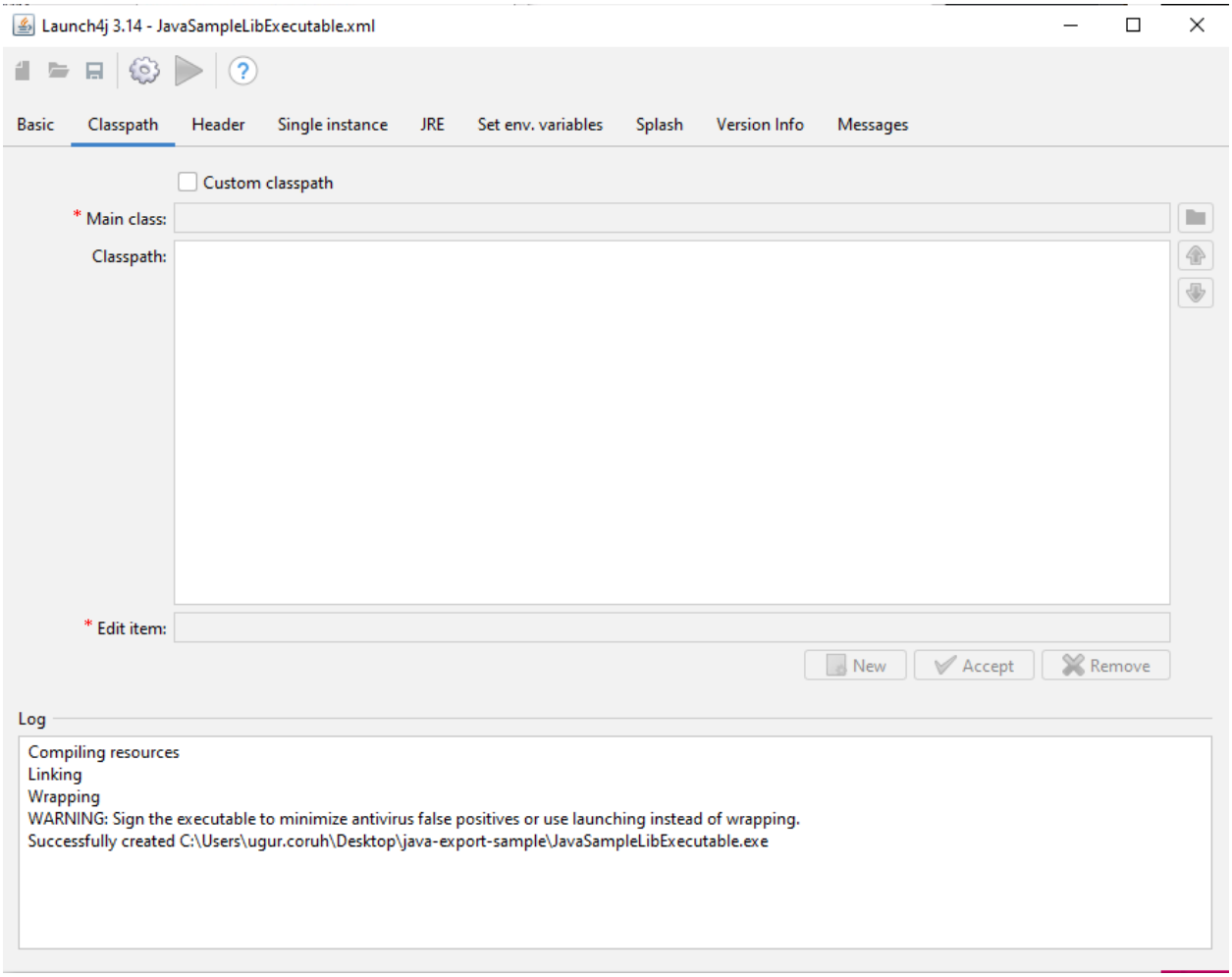
---

configure your application settings similar to below select jar file and exe output path



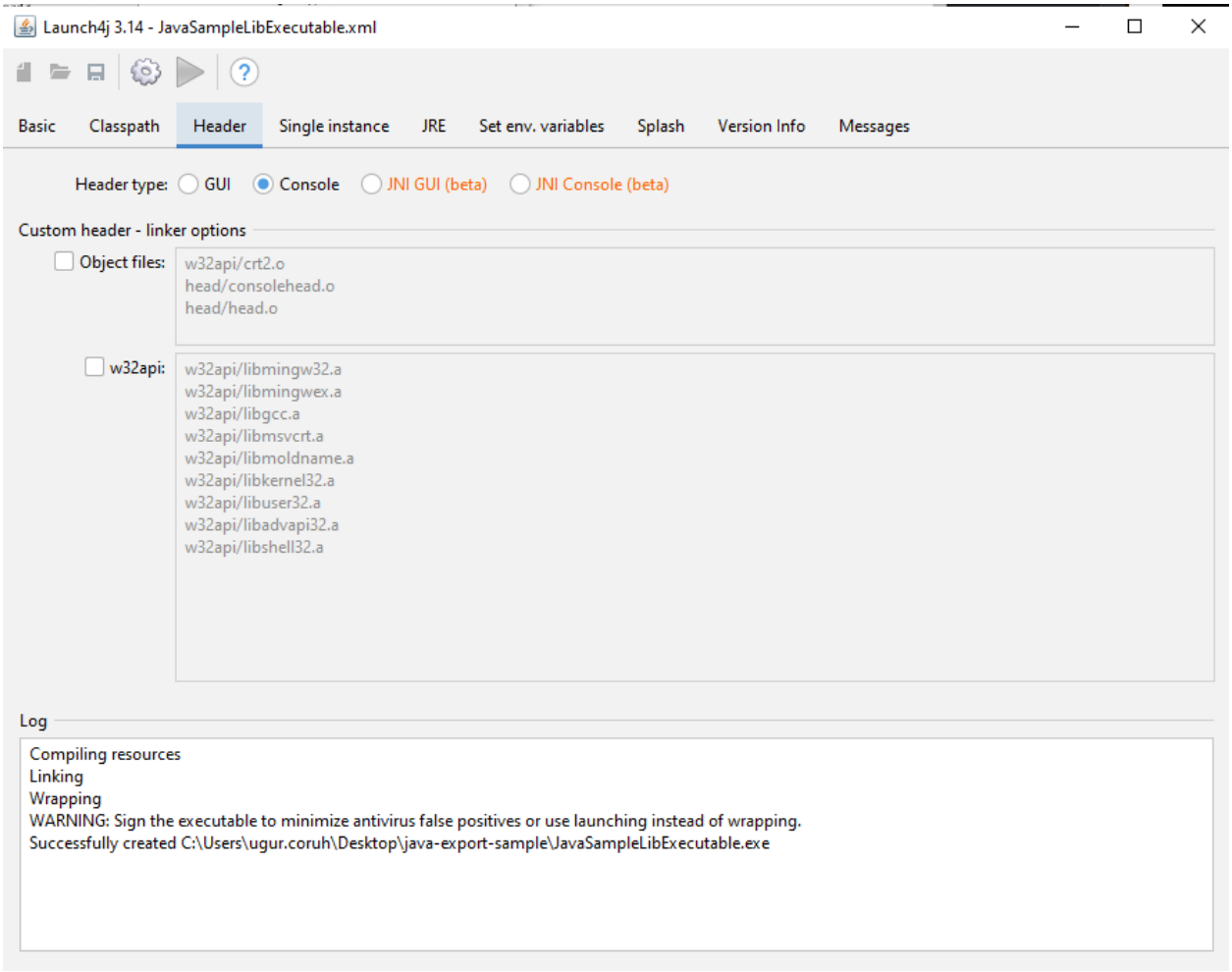
---

we can customize main class if have multiple main class



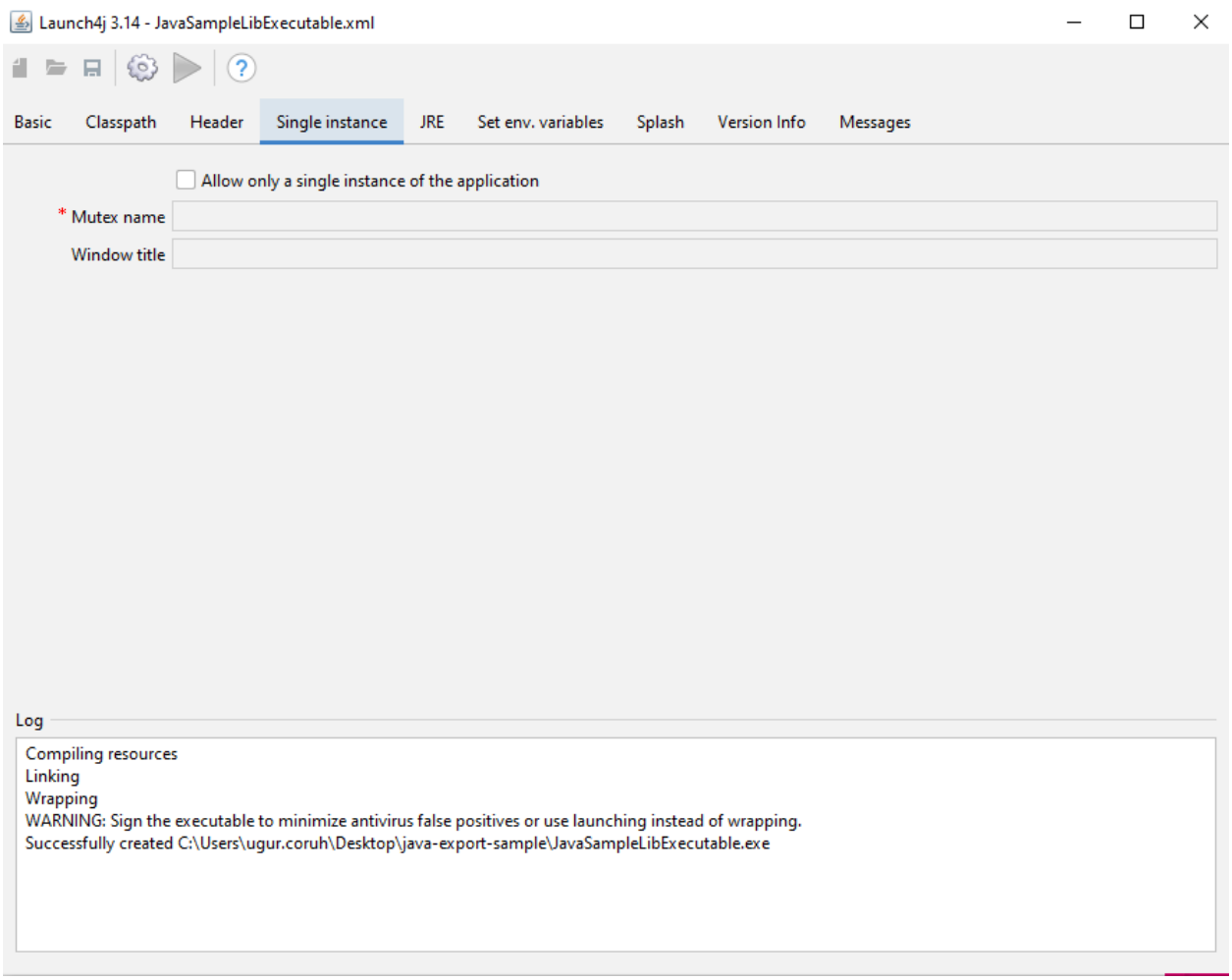
---

select console from setting for this application



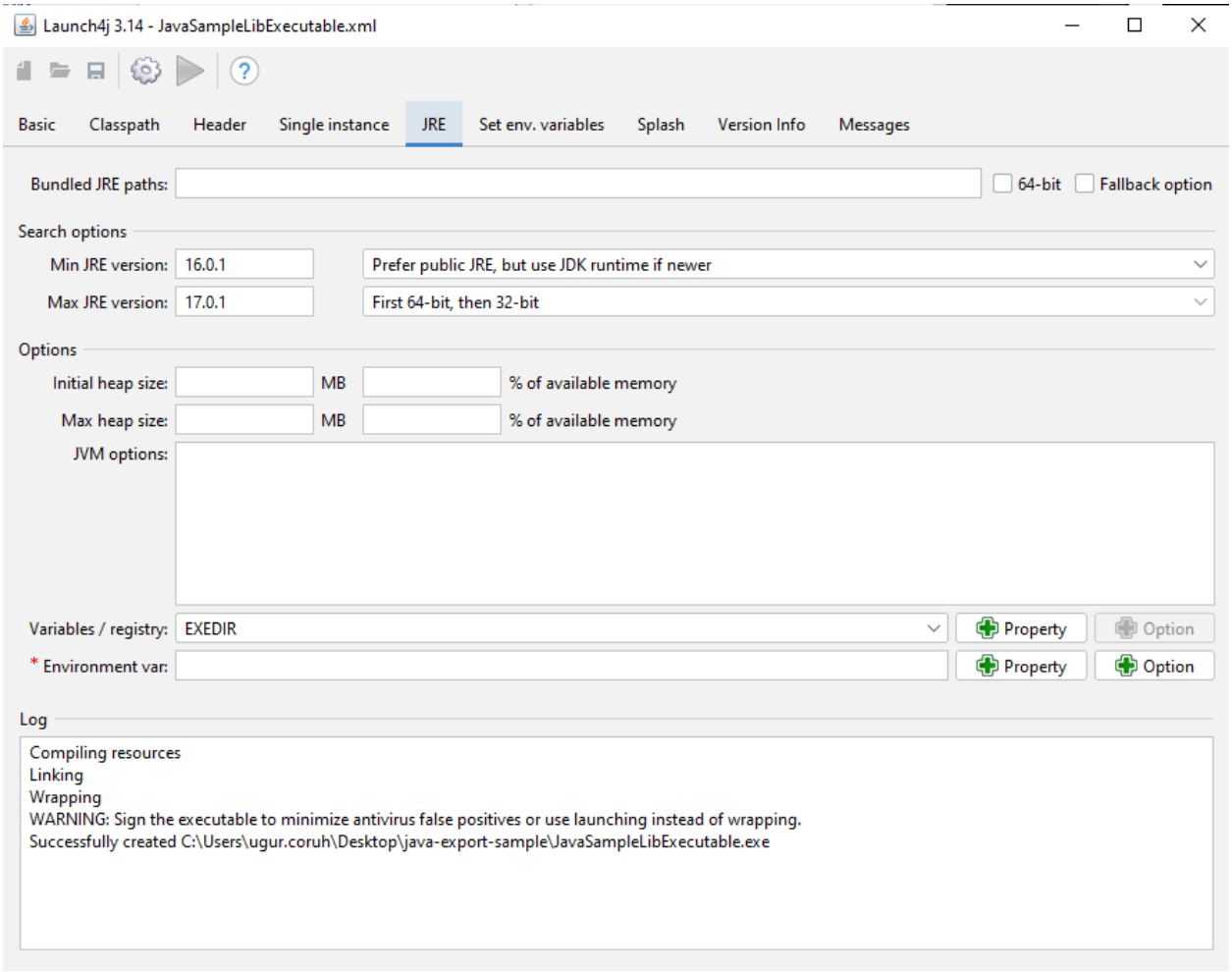
---

we can provide a single running application, this setting avoid to run multiple instances



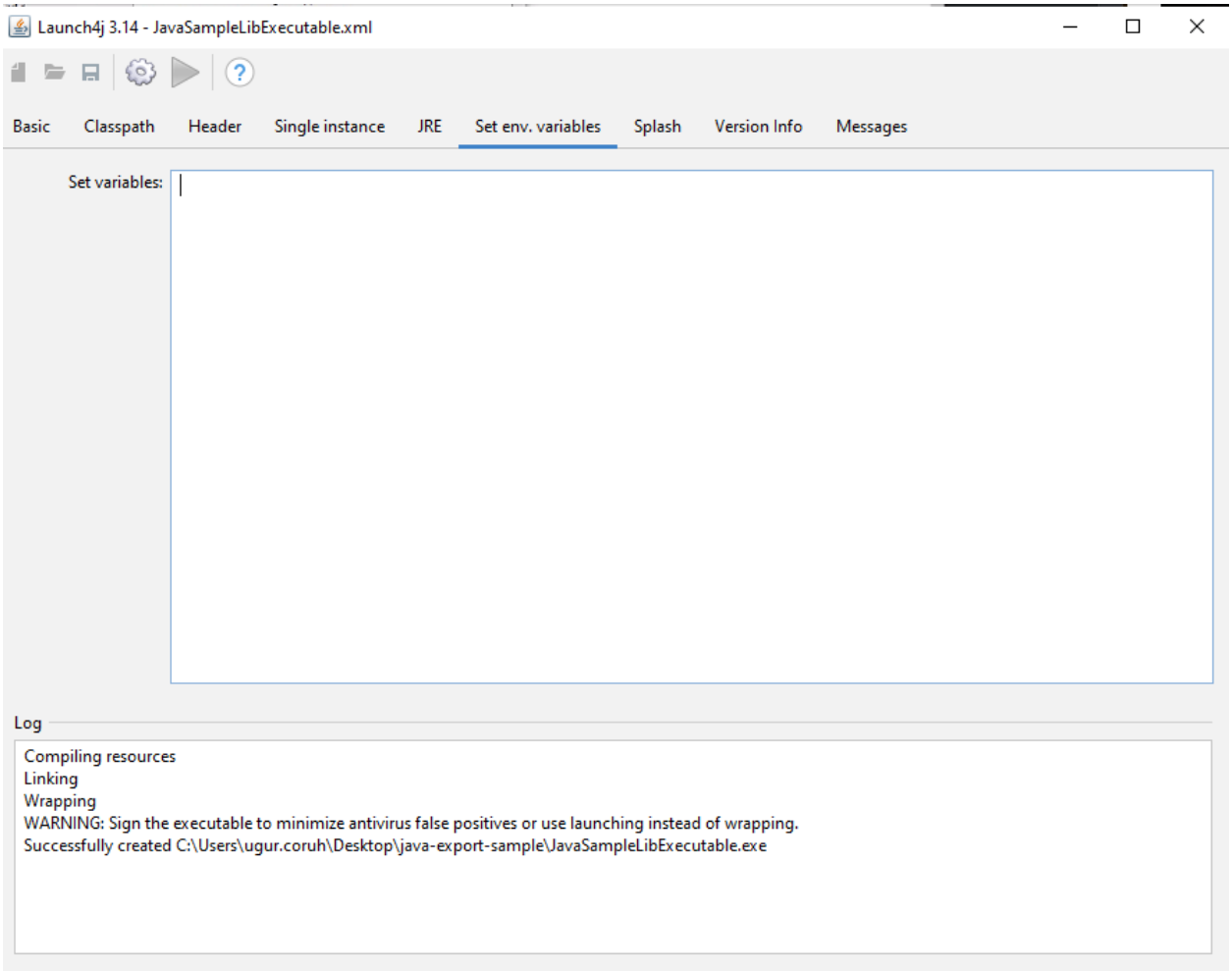
---

we need to set runtime environment versions



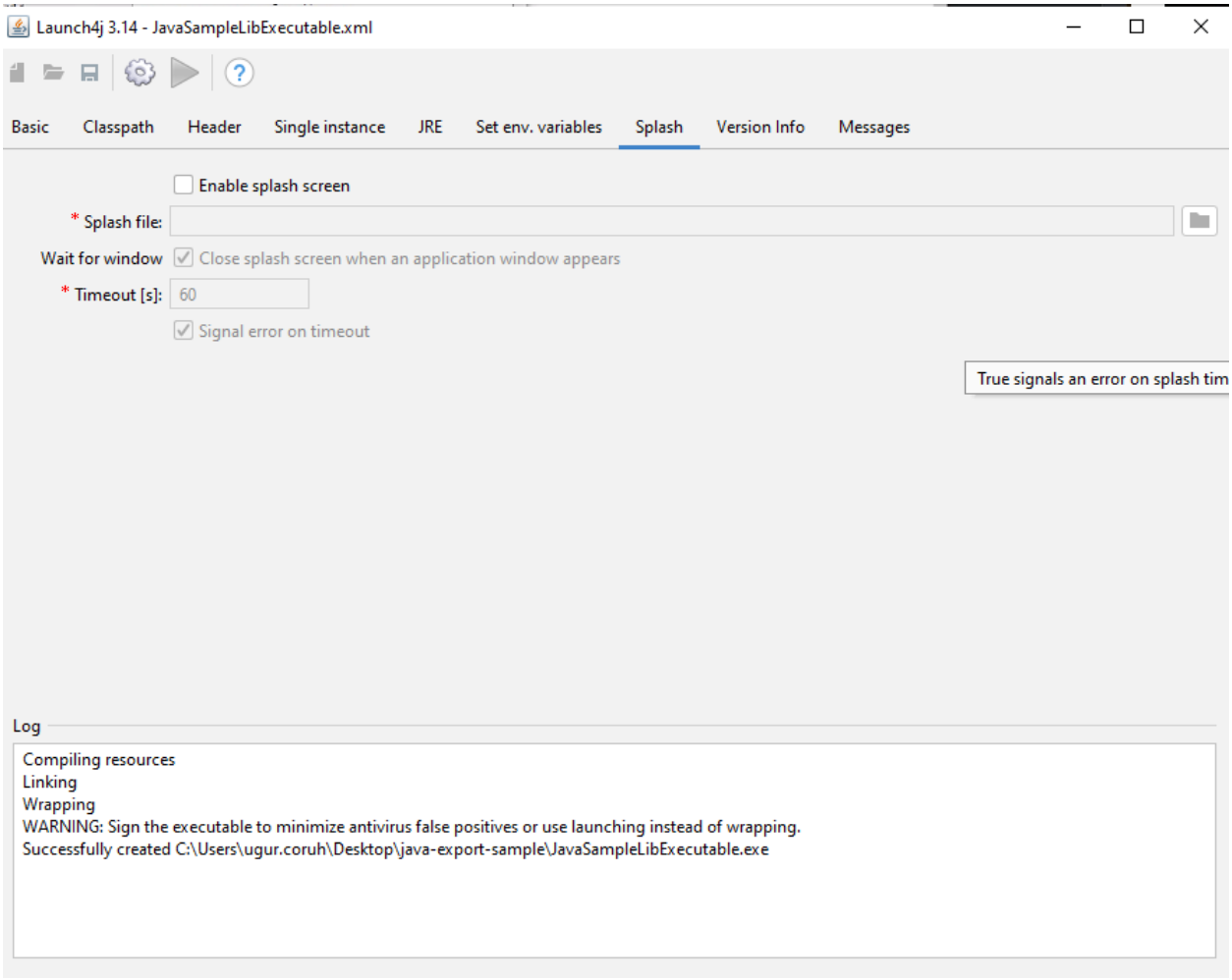
---

you can set system parameters before running application

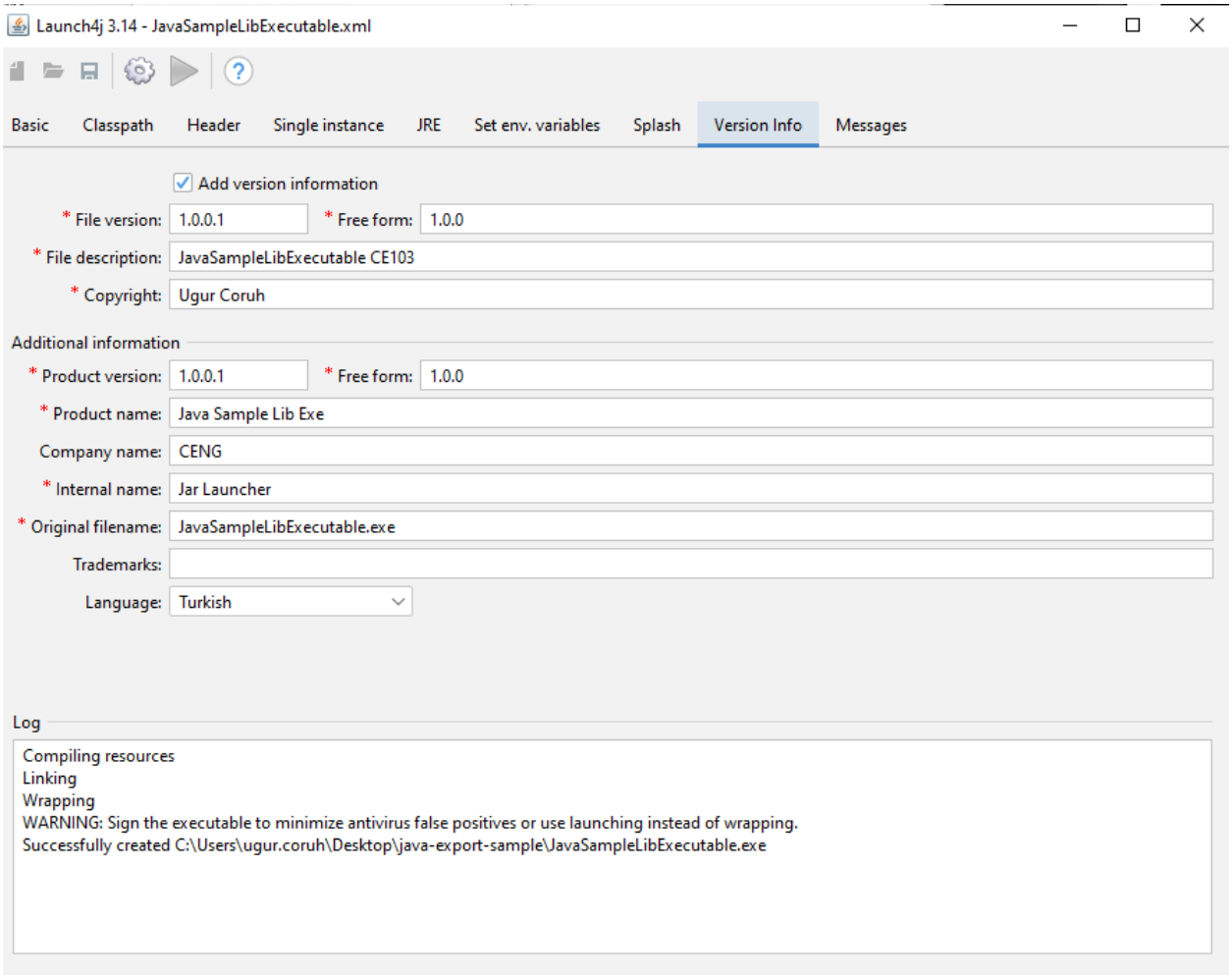


---

with splash screen you can show a splash screen image for your application

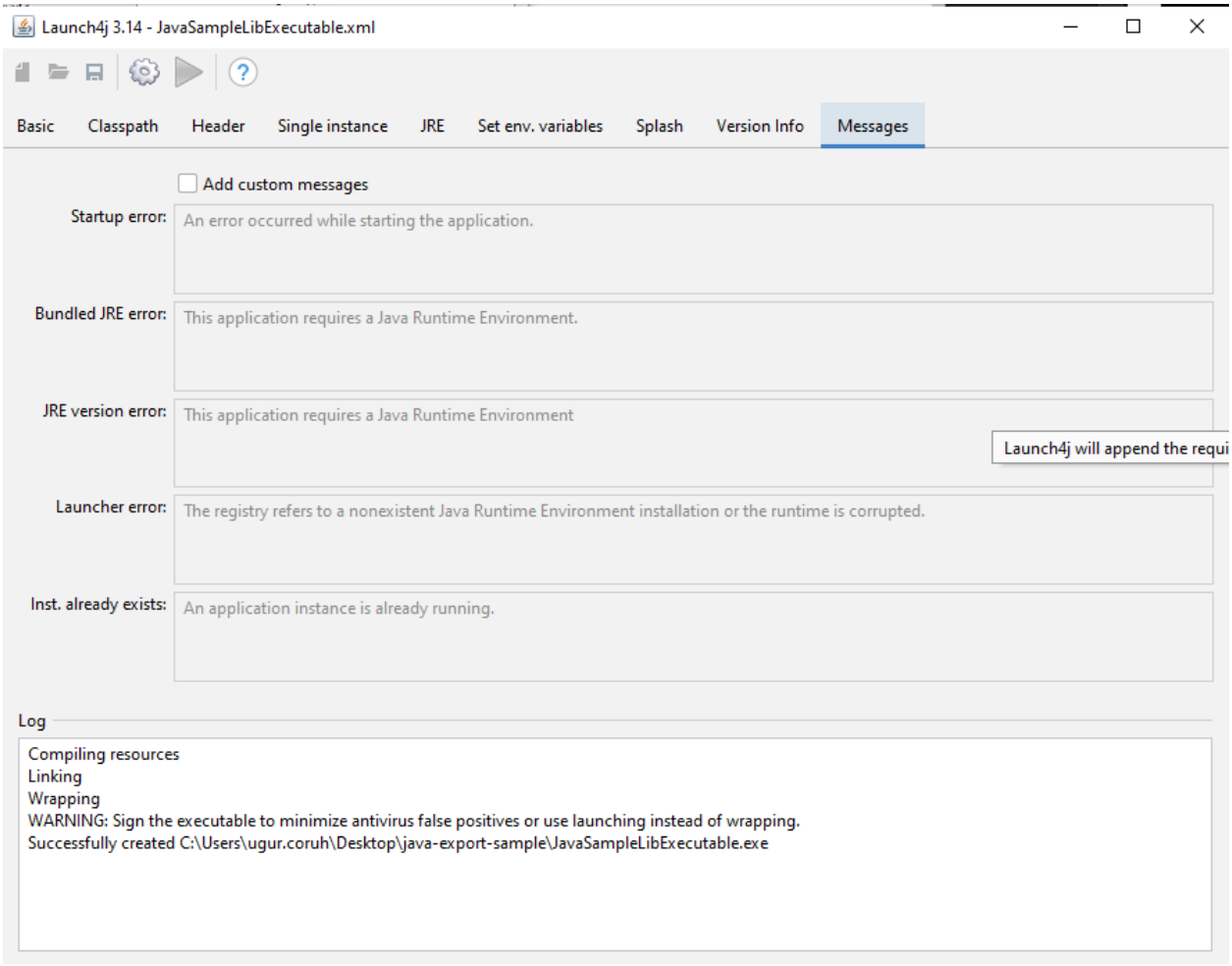


File attributes such as version product information is configured from version info tab



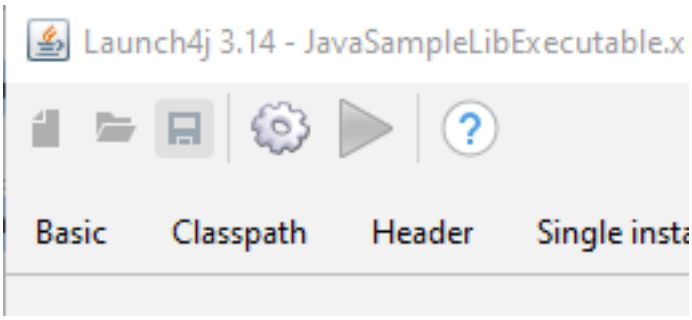
---

if your application runtime condition has an error then you can show this customized messages also



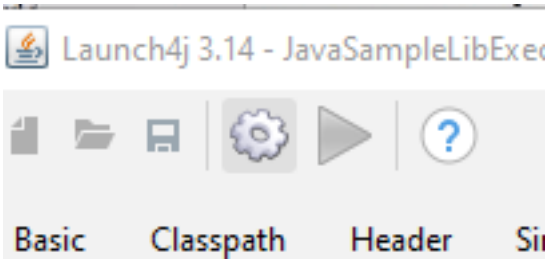
---

with this options save configuration file xml



---

and compile settings



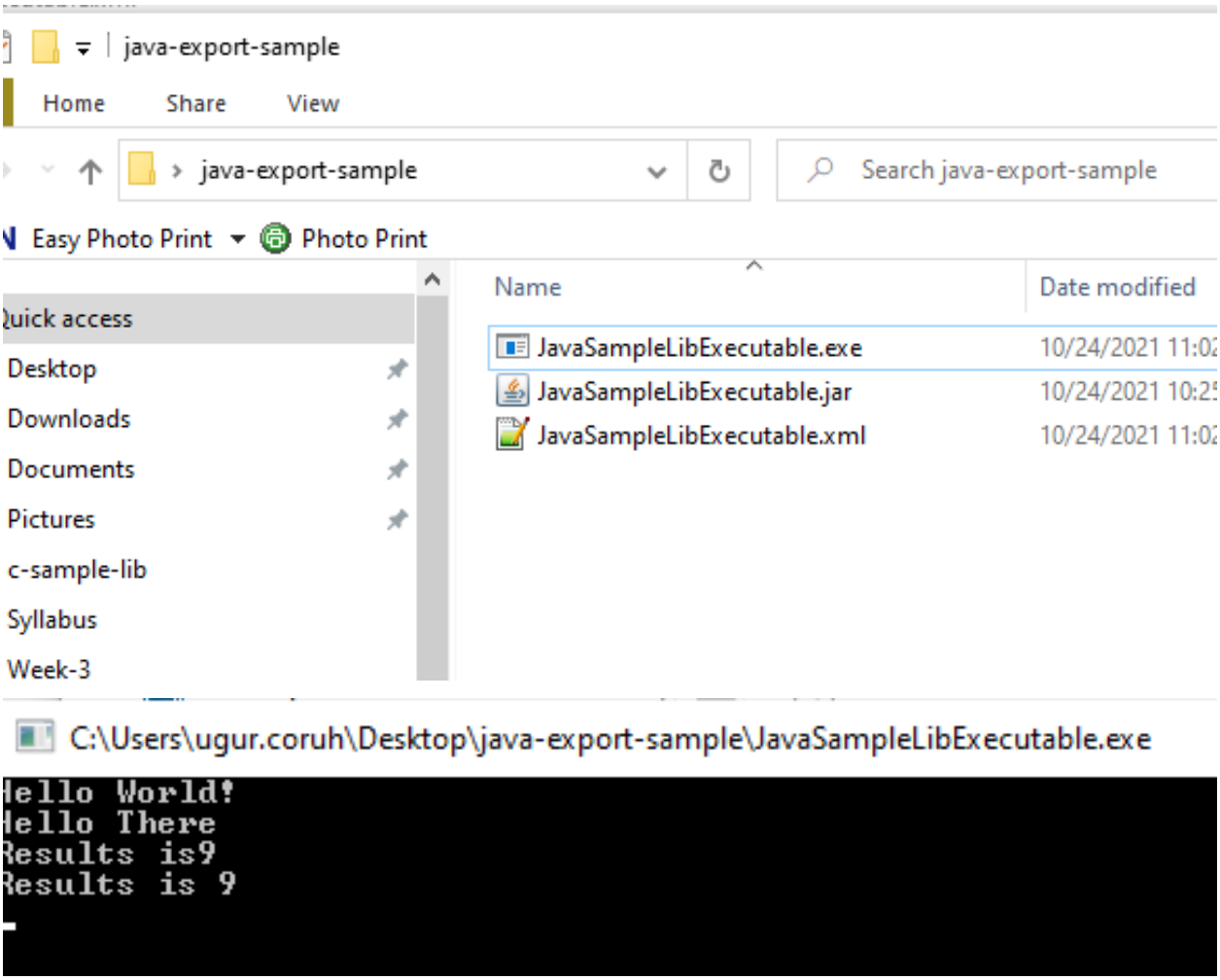
---

you will see generated output file in log screen

```
Compiling resources
Linking
Wrapping
WARNING: Sign the executable to minimize antivirus false positives or use launching instead of wrapping
Successfully created C:\Users\ugur.coruh\Desktop\java-export-sample\JavaSampleLibExecutable.exe
```

---

now we can run exe by click



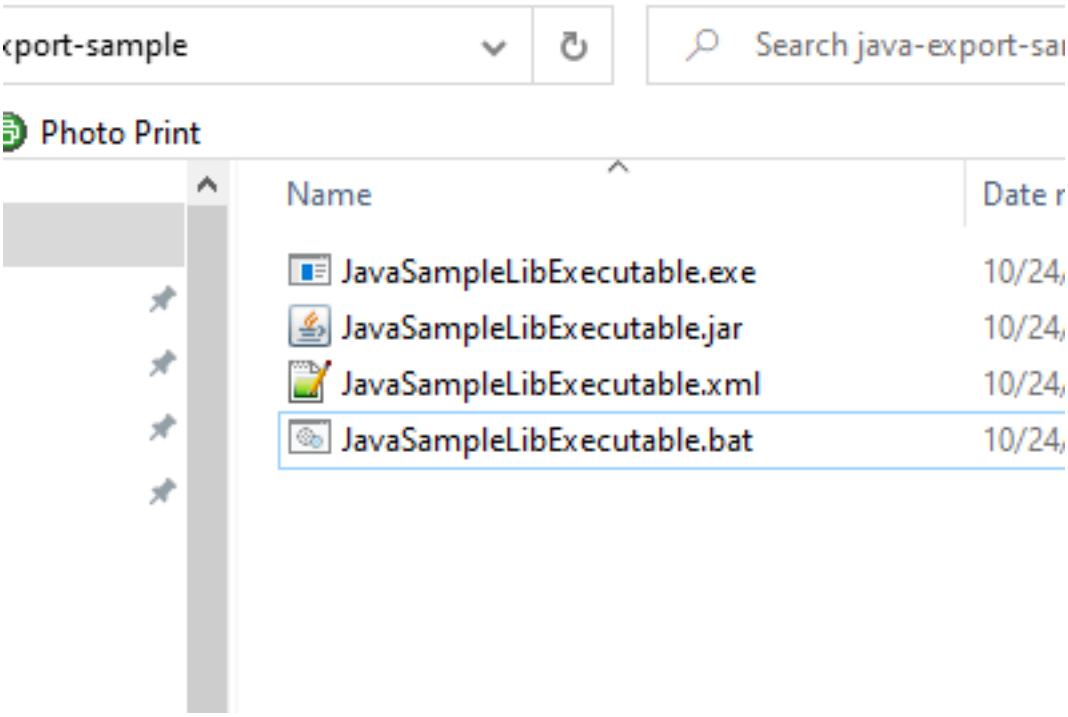
---

another option here adding a bat file to run current jar file

---

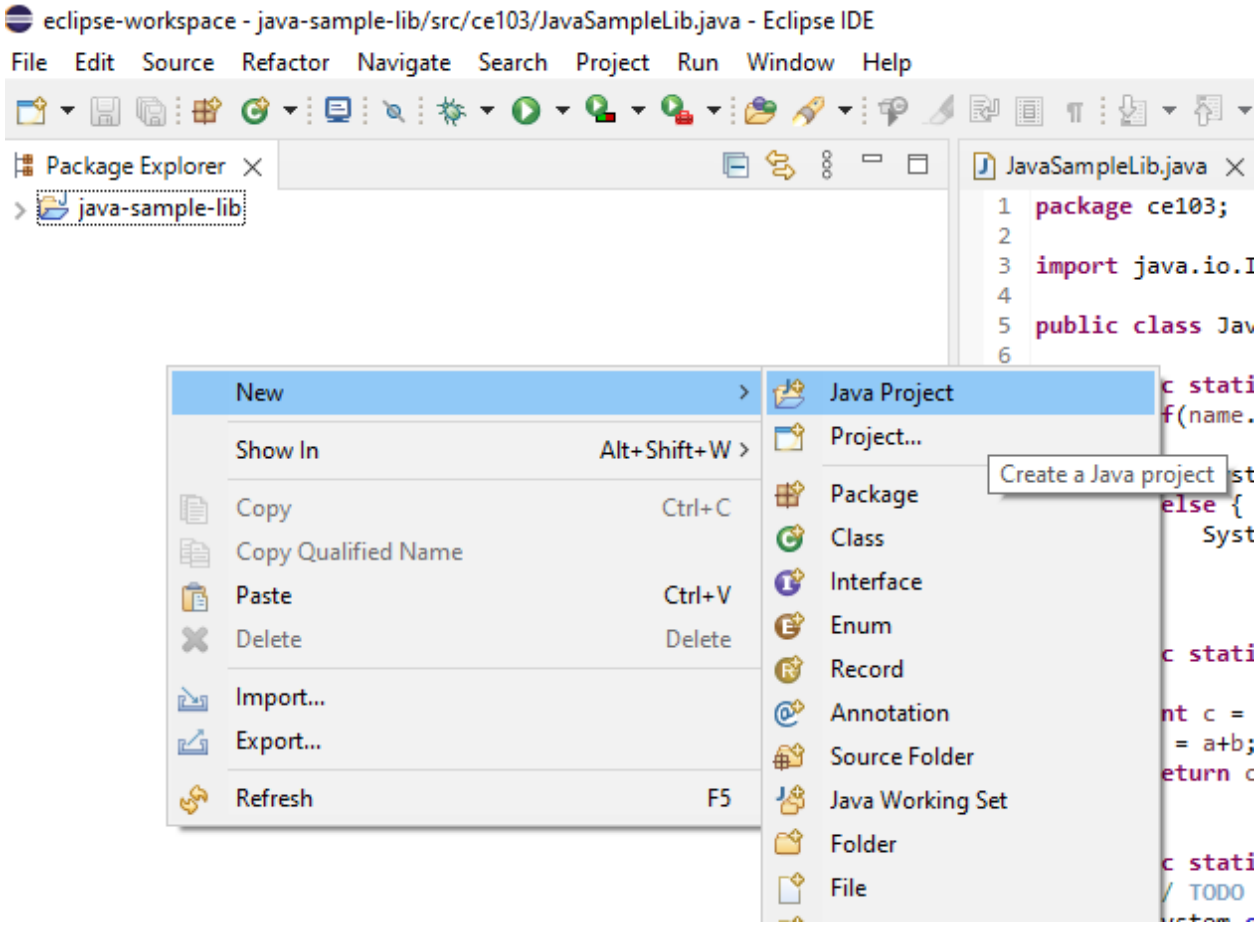
JavaSampleLibExecutable.bat

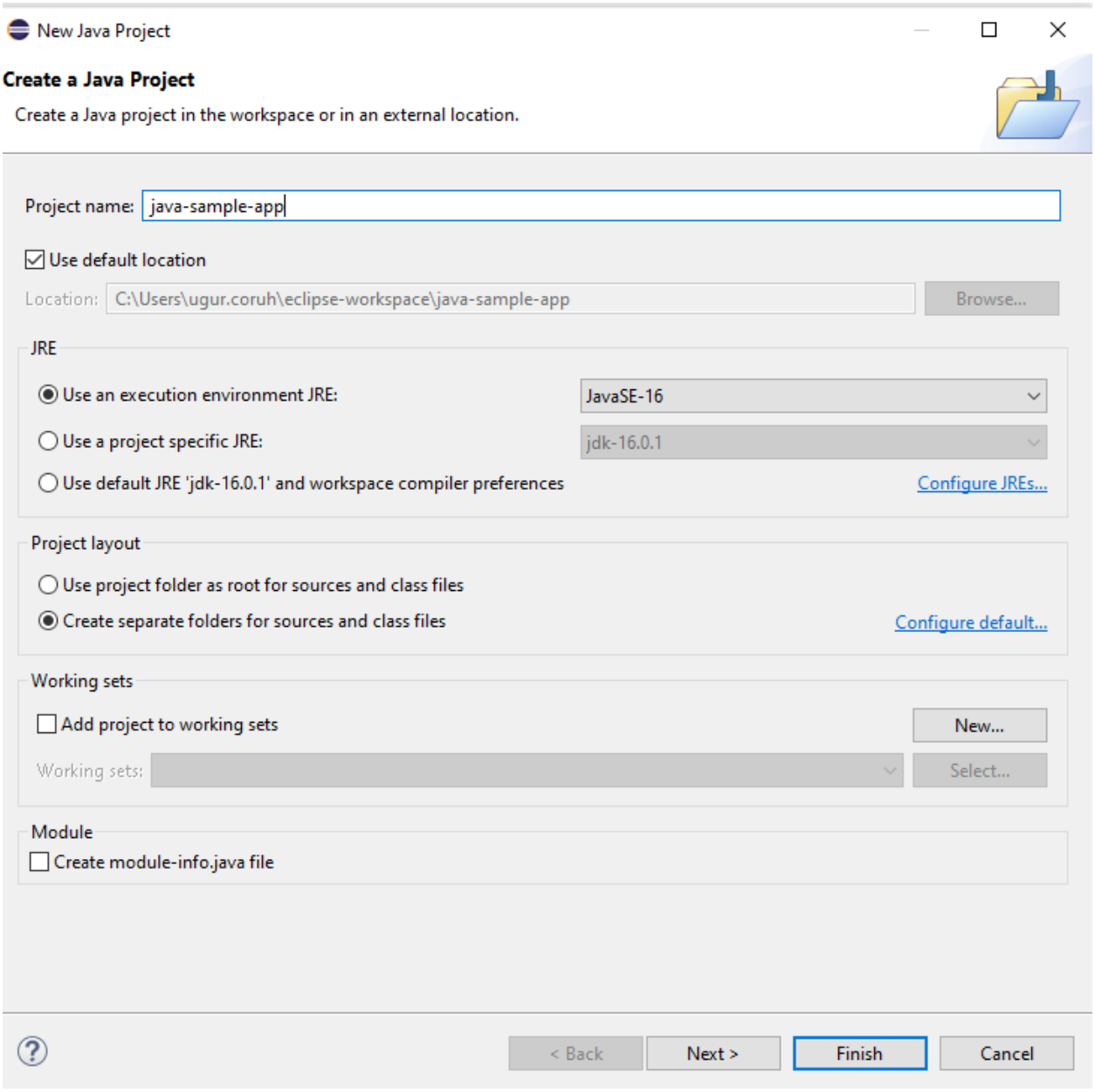
```
java -jar JavaSampleLibExecutable.jar
```

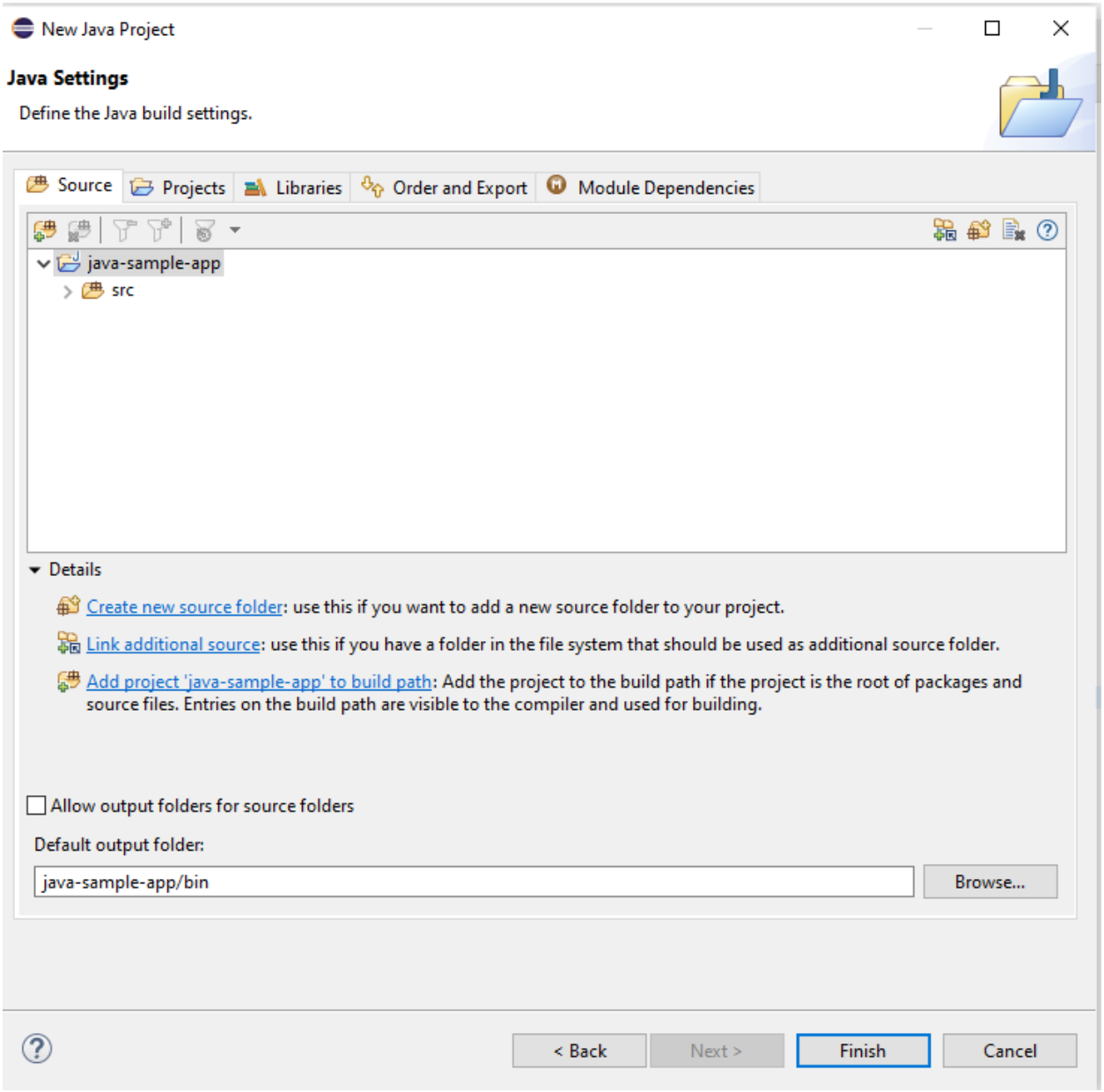


if we click bat file then we will automate command line task for current jar file

Now return back to our java library and create another console application that use library functions

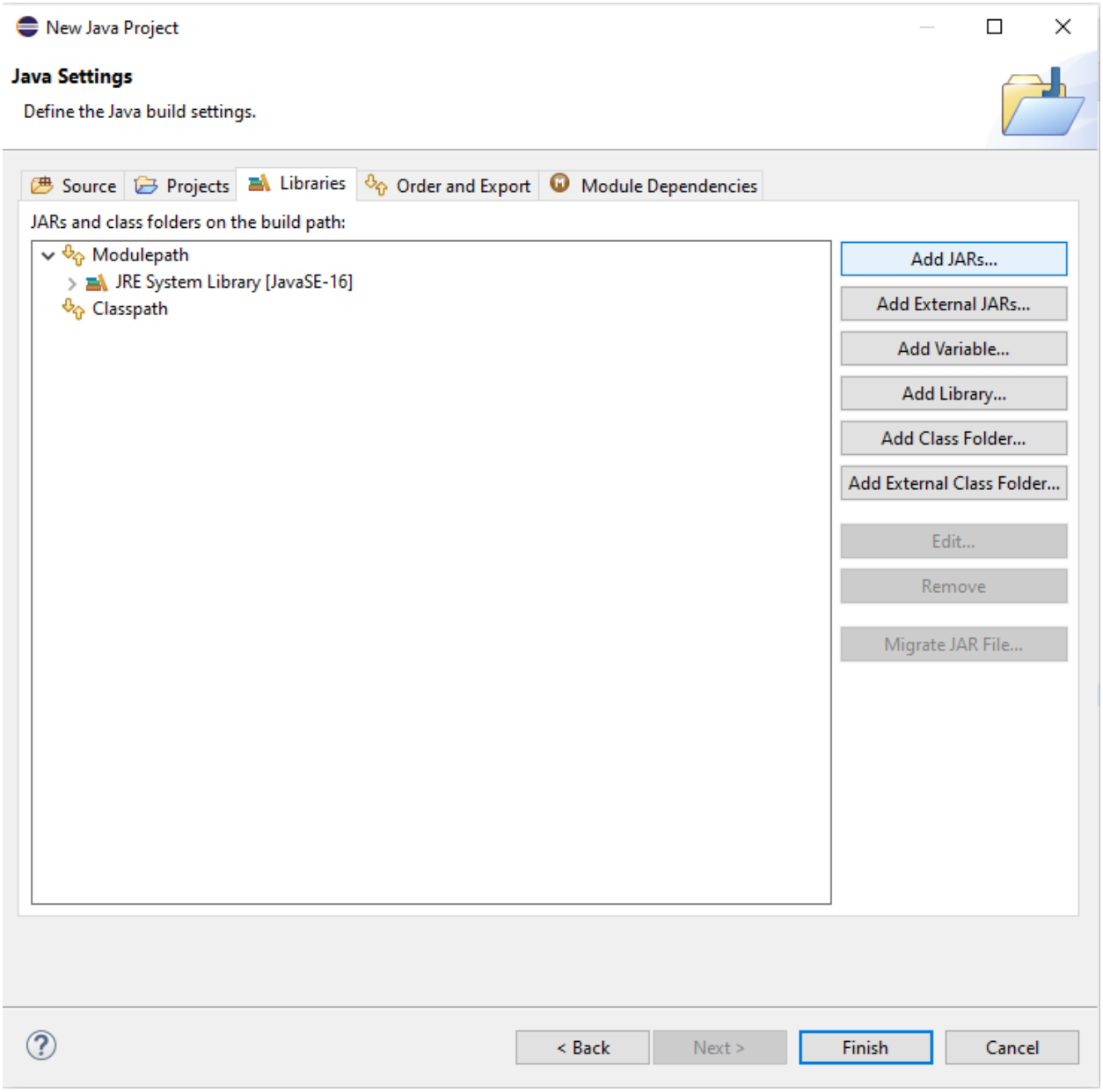

eclipse-workspace - java-sample-lib/src/ce103/JavaSampleLib.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
java-sample-lib
JavaSampleLib.java
1 package ce103;
3 import java.io.I
5 public class Jav
New
Show In Alt+Shift+W
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Import...
Export...
Refresh F5
Java Project
Project...
Package
Class
Interface
Enum
Record
Annotation
Source Folder
Java Working Set
Folder
File
Create a Java project


New Java Project
Create a Java Project
Create a Java project in the workspace or in an external location.
Project name: java-sample-app
Use default location
Location: C:\Users\ugur.coruh\eclipse-workspace\java-sample-app
Browse...
JRE
Use an execution environment JRE:
JavaSE-16
Use a project specific JRE:
jdk-16.0.1
Use default JRE 'jdk-16.0.1' and workspace compiler preferences
Configure JREs...
Project layout
Use project folder as root for sources and class files
Create separate folders for sources and class files
Configure default...
Working sets
Add project to working sets
New...
Working sets:
Select...
Module
Create module-info.java file
< Back
Next >
Finish
Cancel

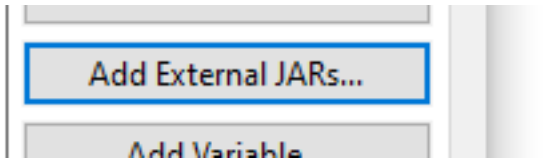


---

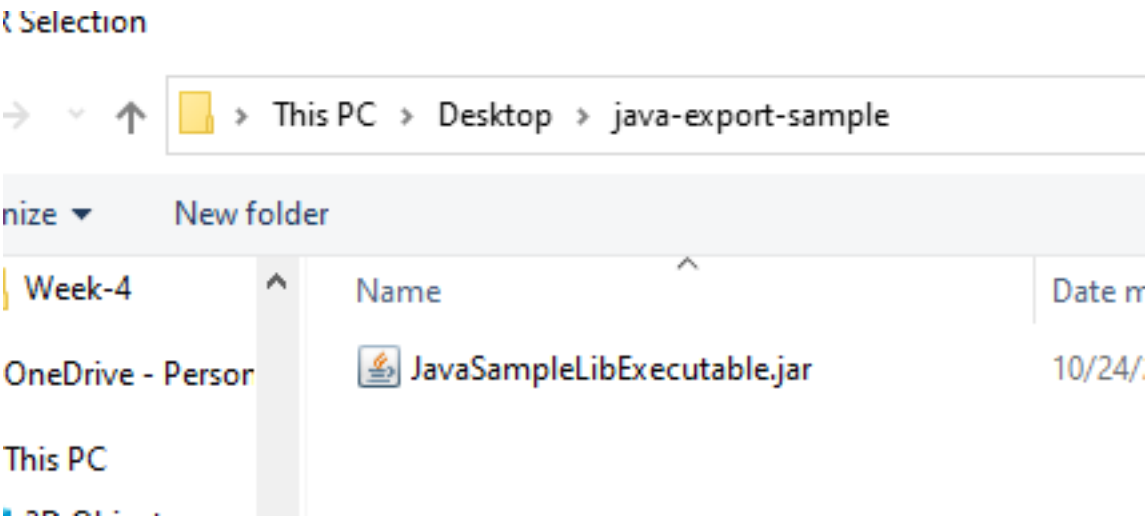
you can set libraries in this step from but our library should exported for our solution



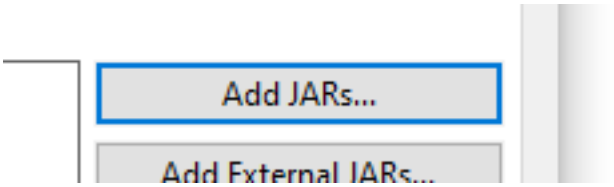
Select Add External JARs...

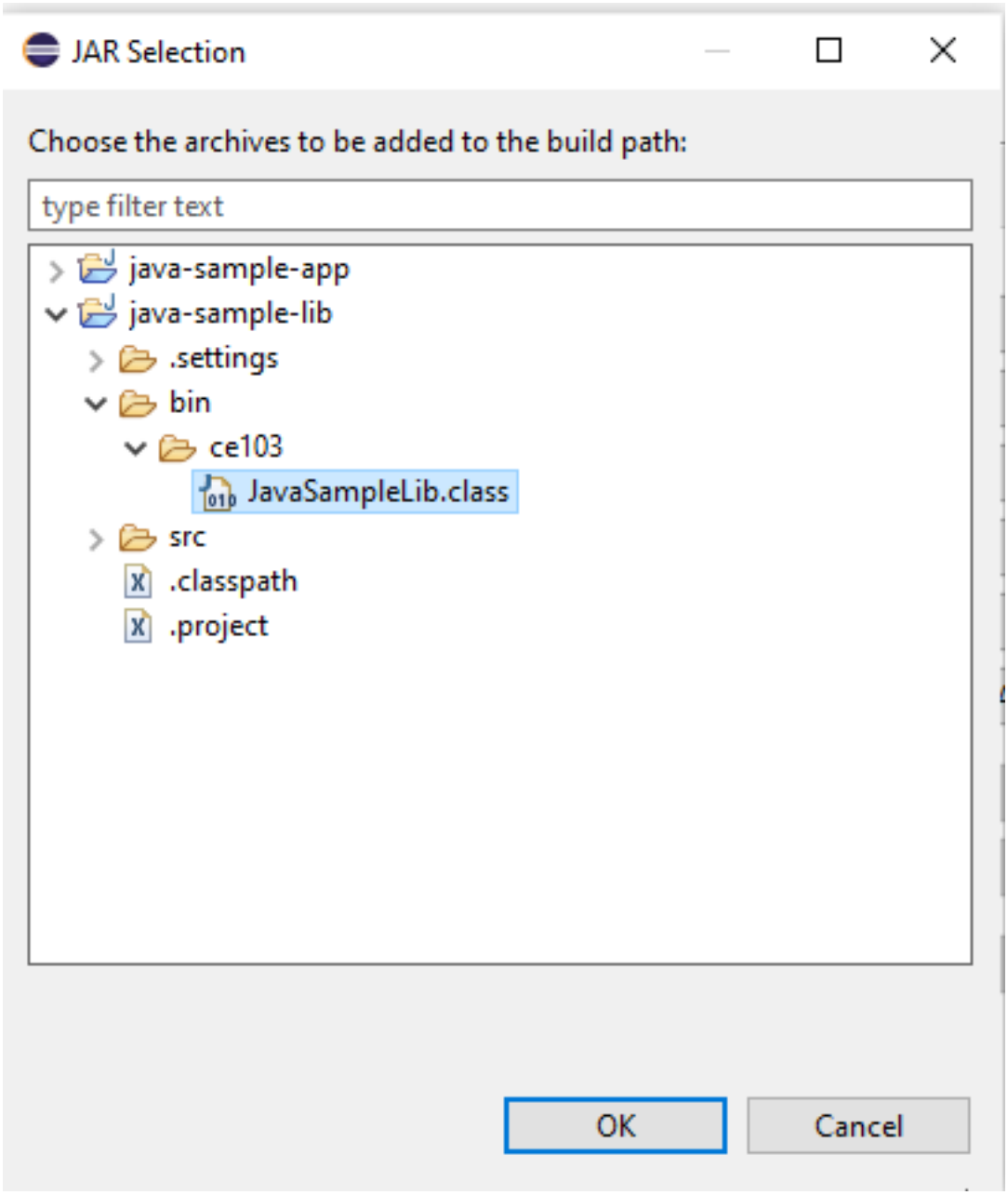


Open Exported jar folder and select



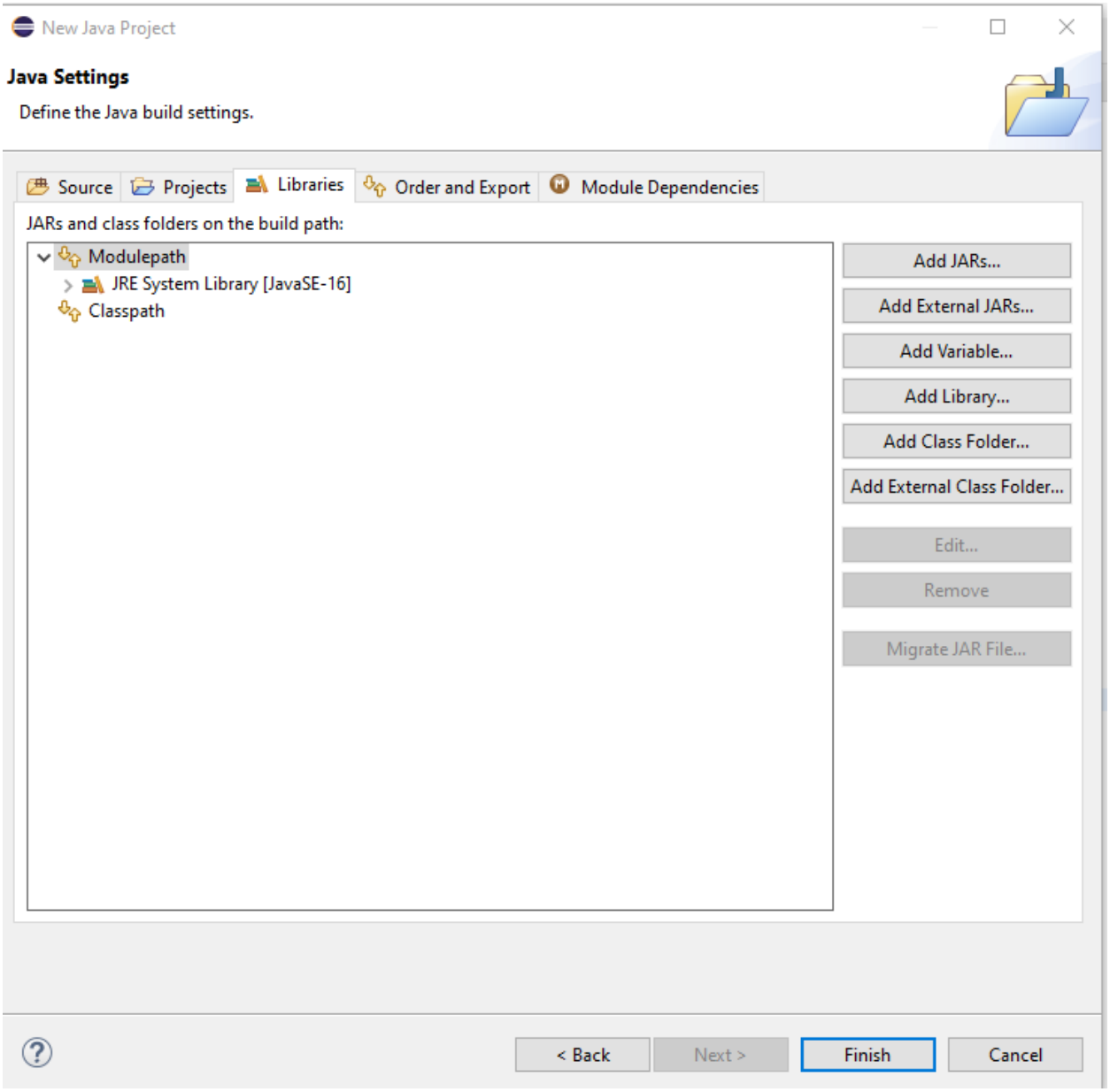
Or we can select by Add jar from current workspace



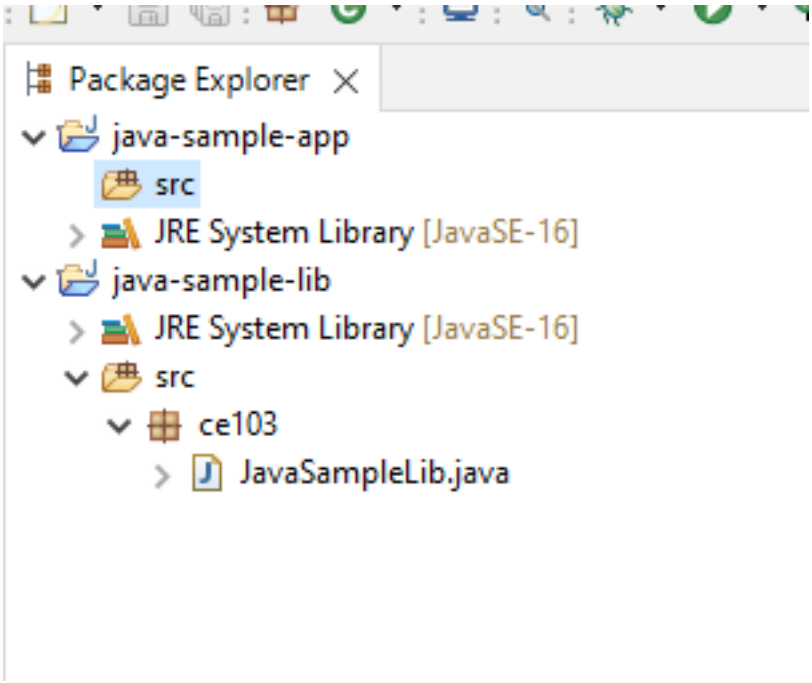


---

but in this step I won't add anything I'll add references later

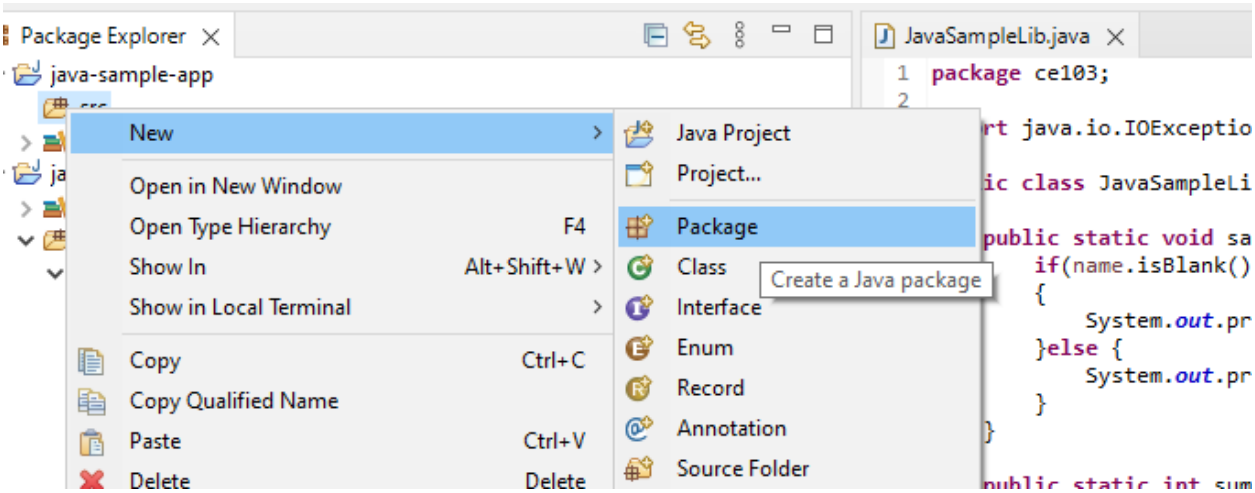


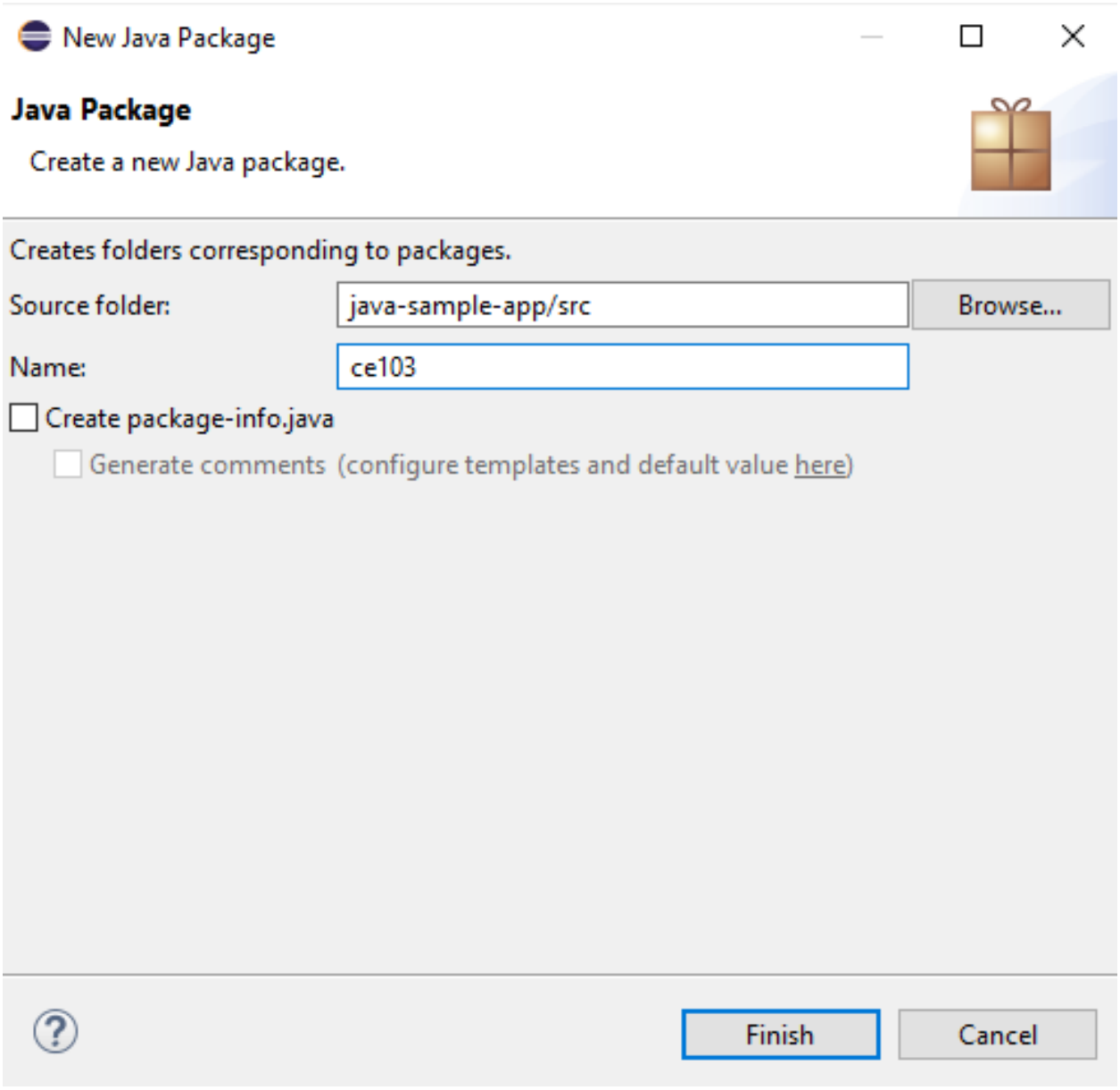
we will have the following project



---

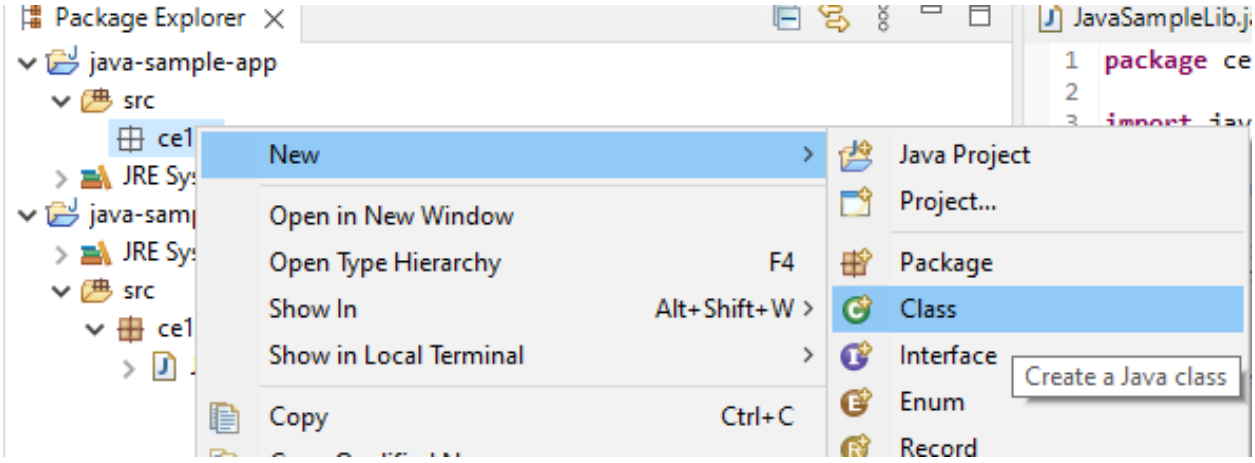
lets create a package



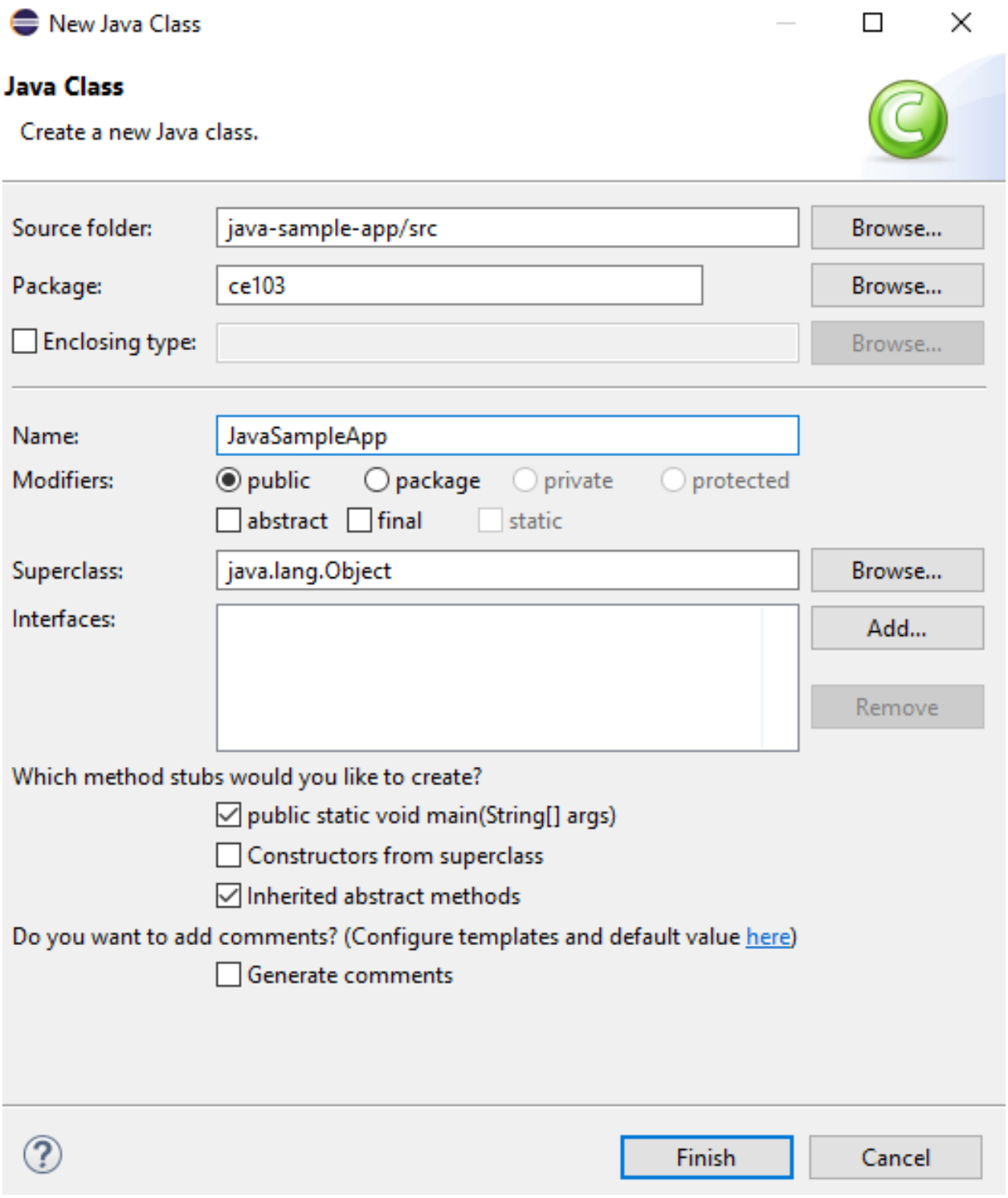


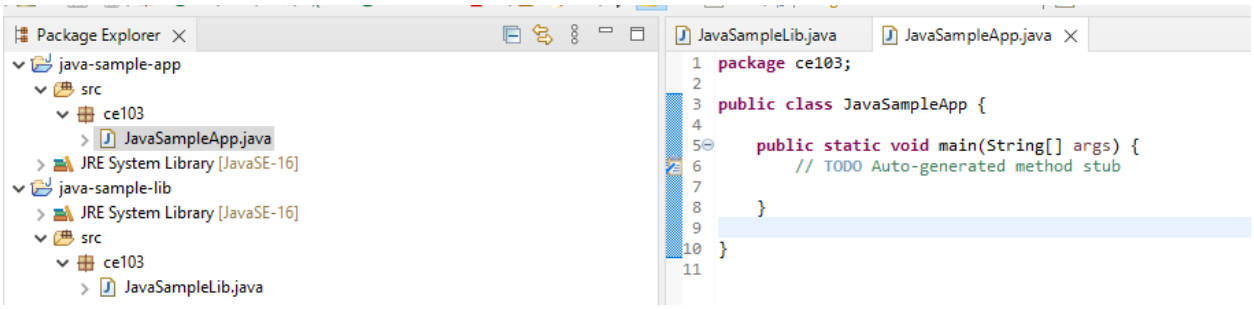
---

and lets create a main class for our application

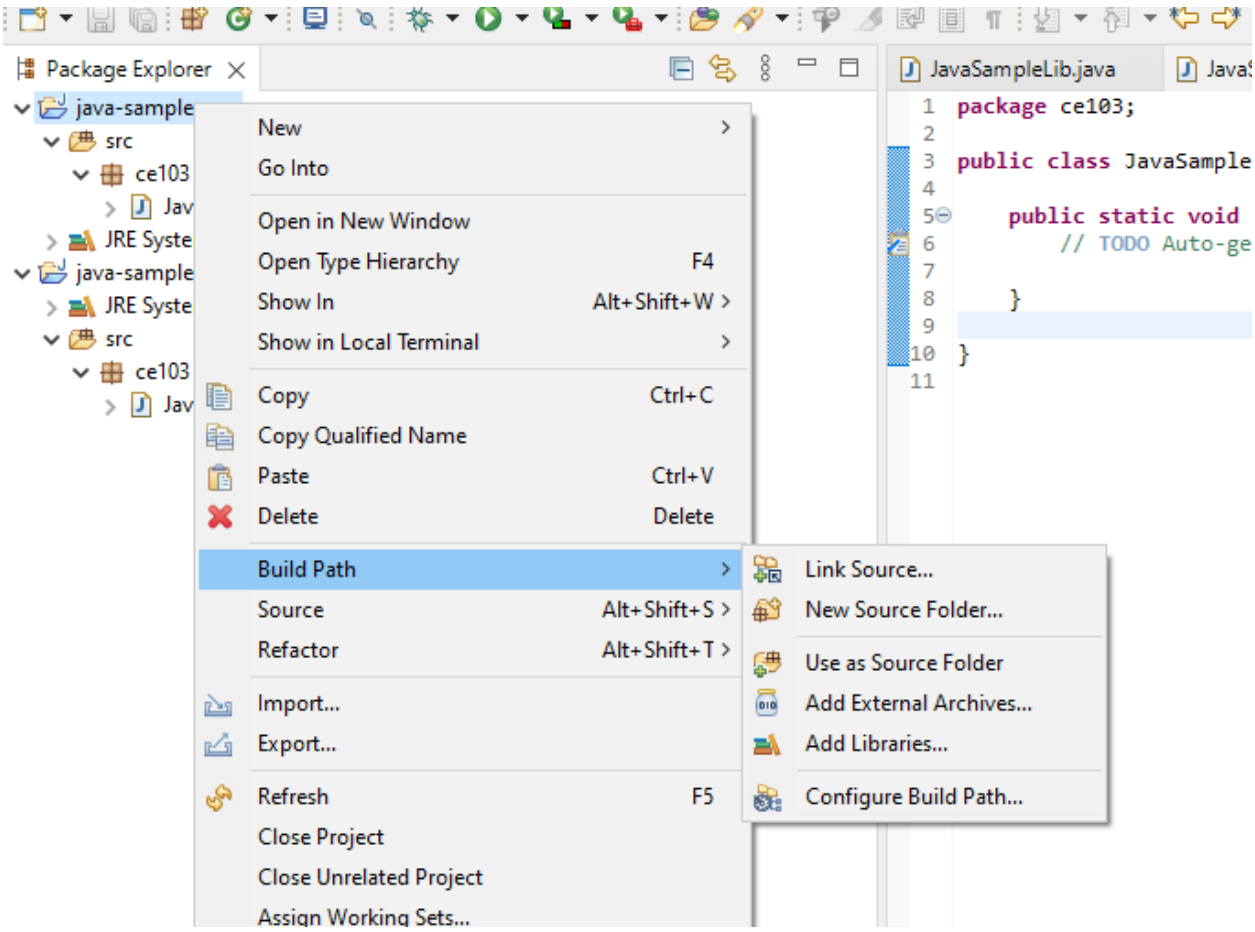


check create main function

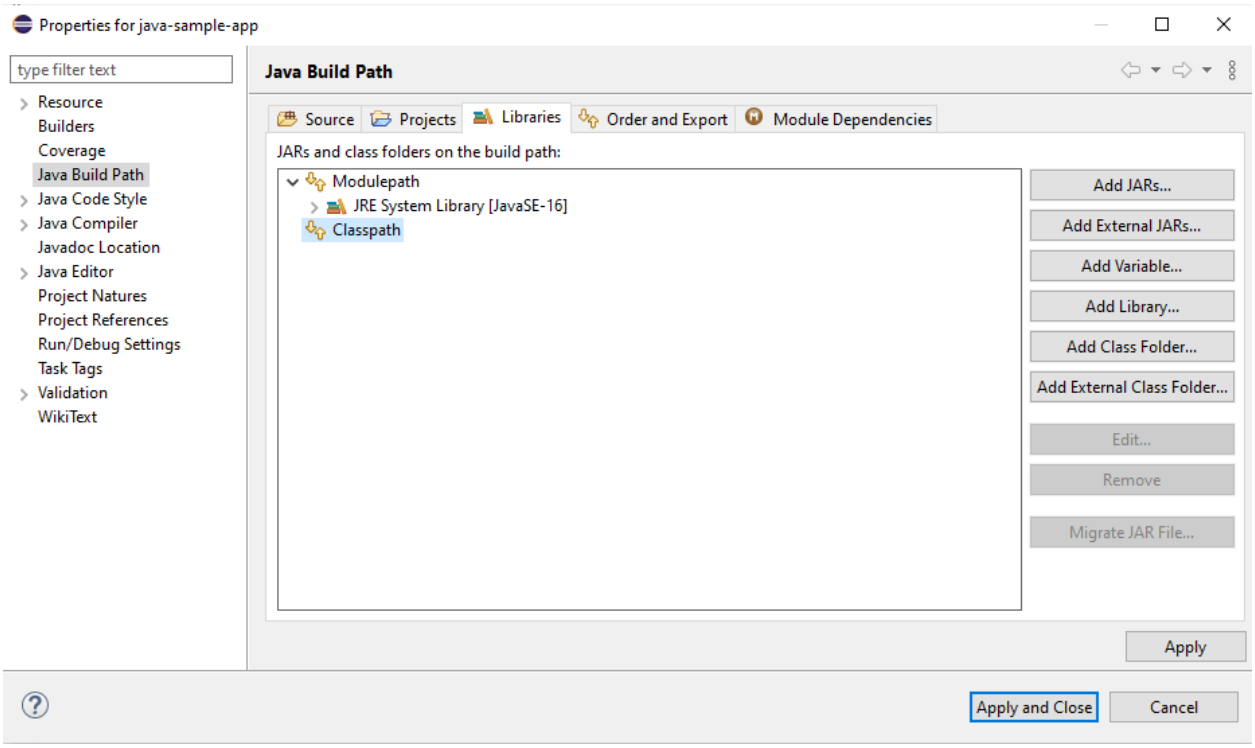




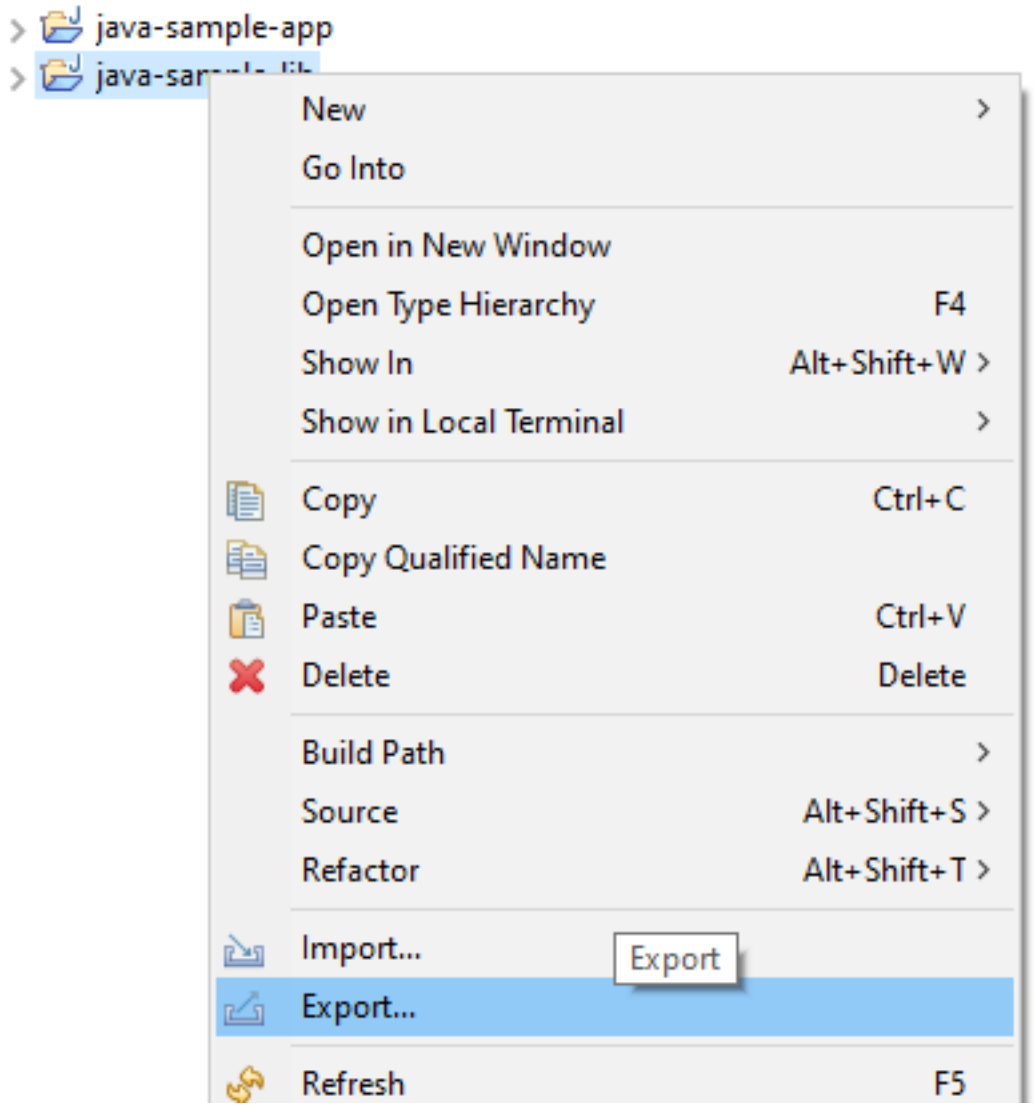
right click to project and add reference



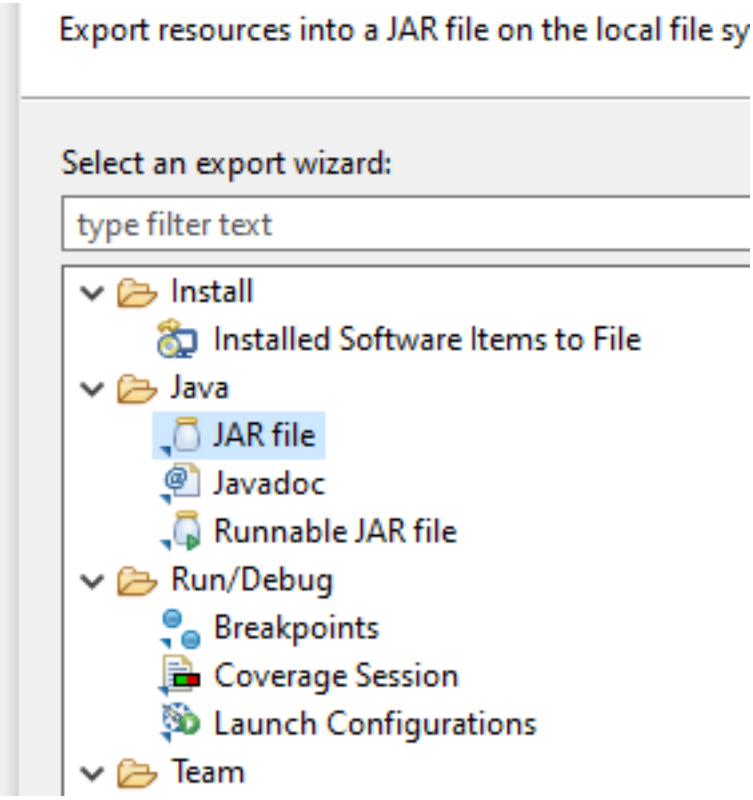
you can enter same configurations from project properties



Lets export our library as a JAR file and then add to our classpath



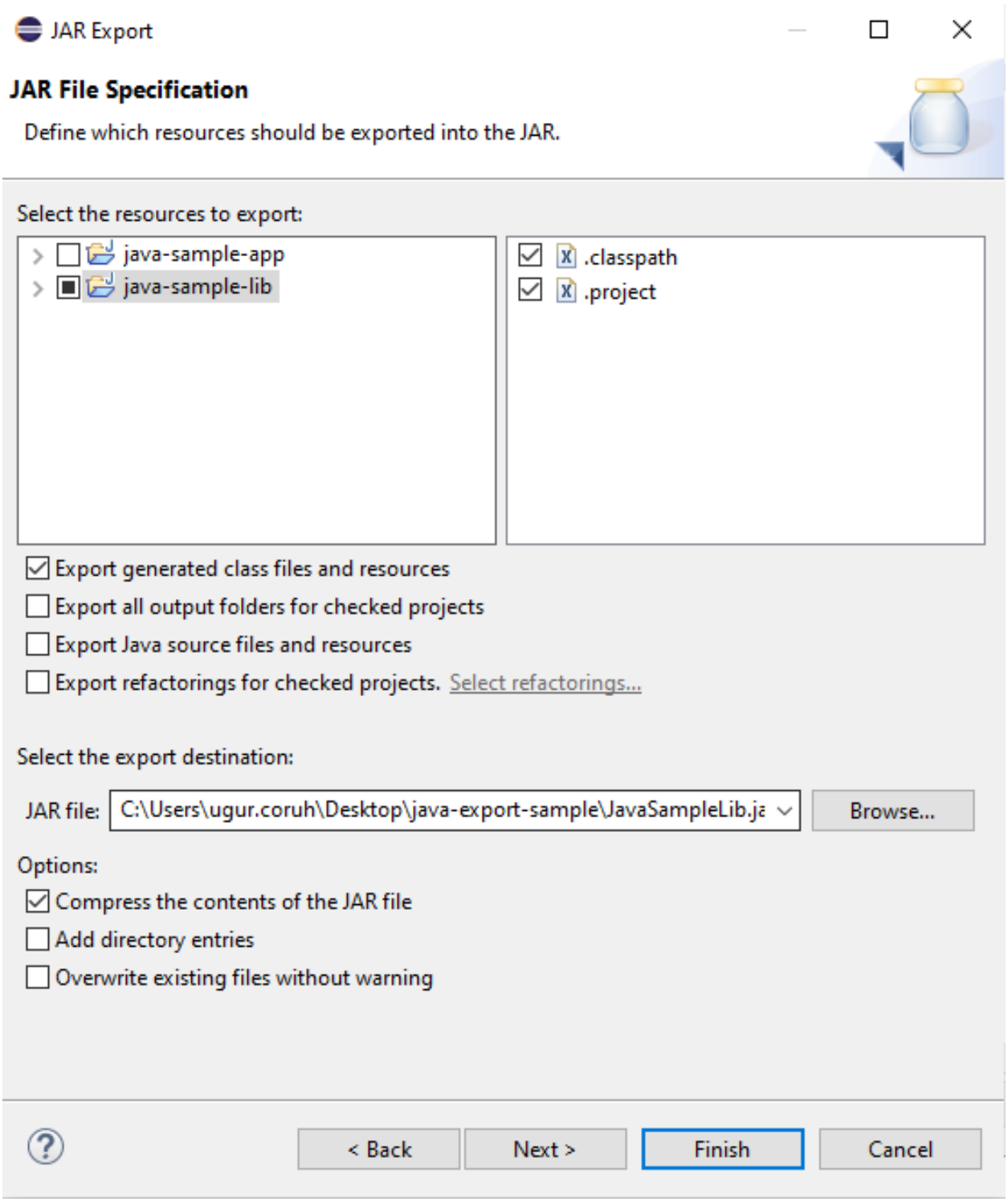
Select JAR file

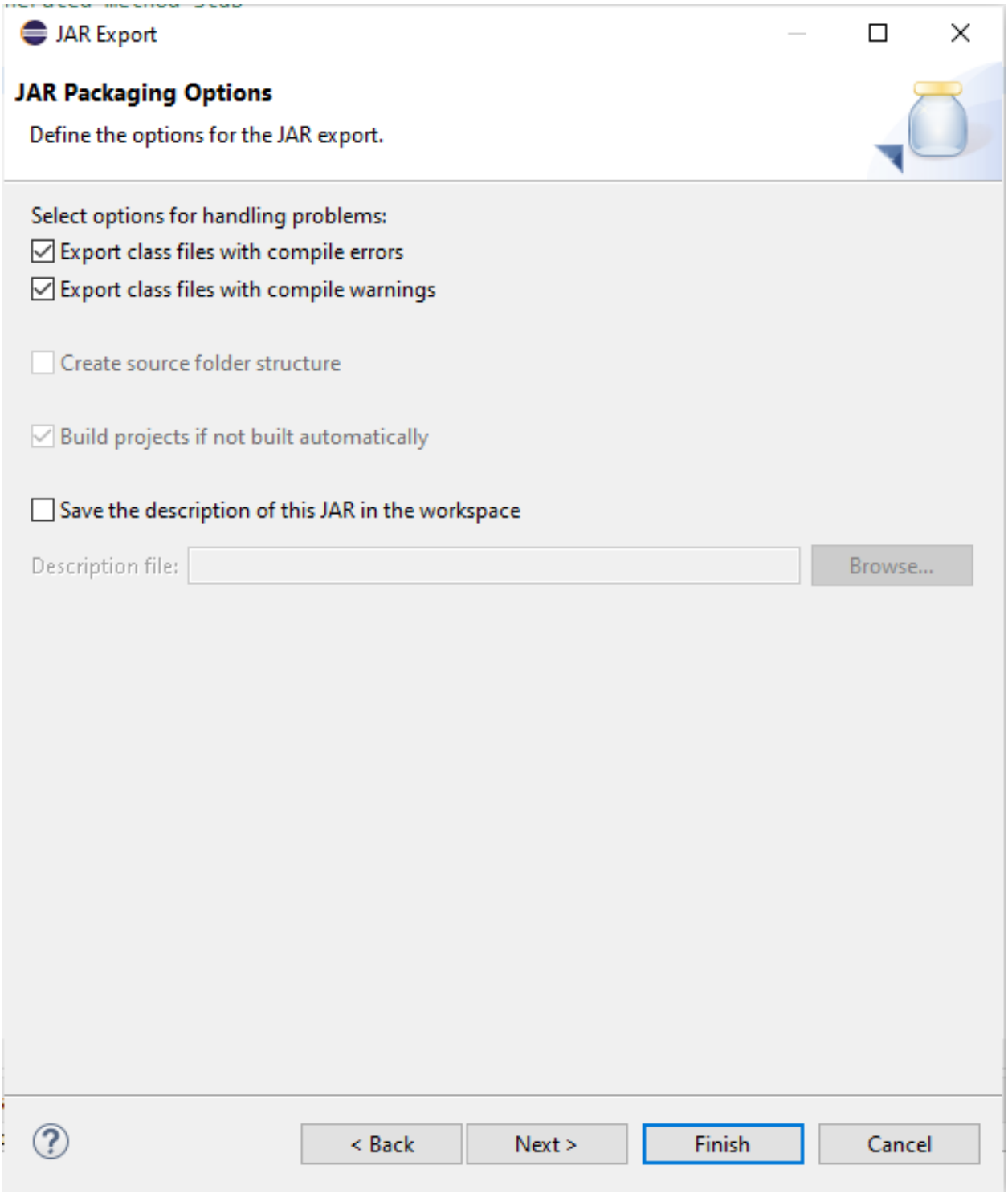


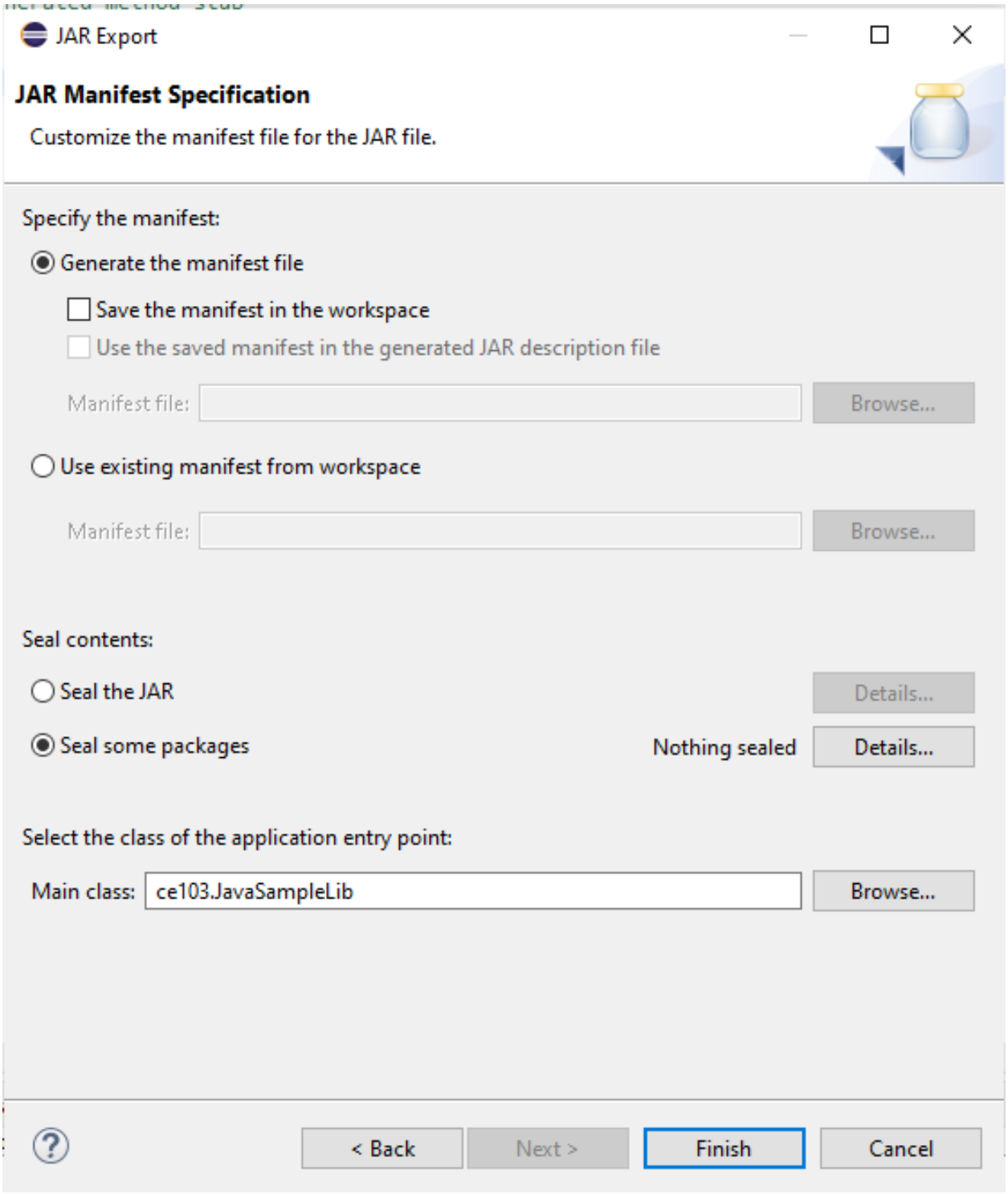
---

we configured output as

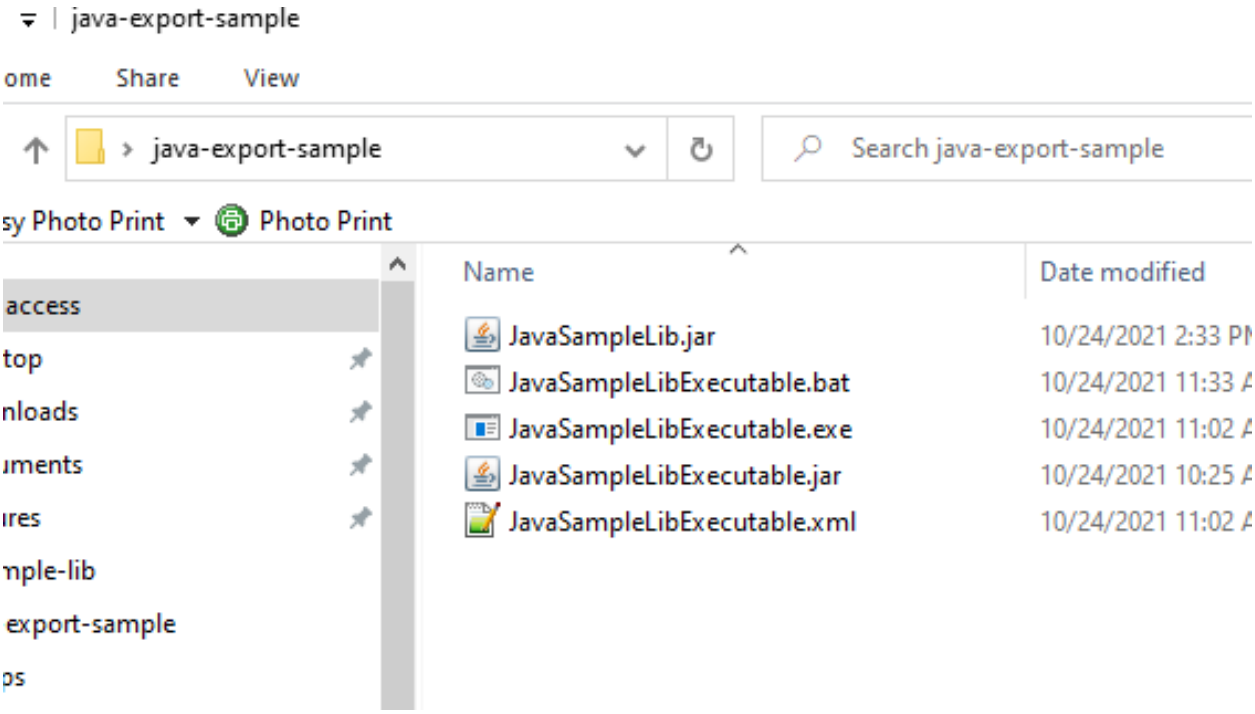
`C:\Users\ugur.coruh\Desktop\java-export-sample\JavaSampleLib.jar`


JAR Export
JAR File Specification
Define which resources should be exported into the JAR.
Select the resources to export:
java-sample-app
java-sample-lib
.classpath
.project
Export generated class files and resources
Export all output folders for checked projects
Export Java source files and resources
Export refactorings for checked projects. Select refactorings...
Select the export destination:
JAR file:
C:\Users\ugur.coruh\Desktop\java-export-sample\JavaSampleLib.ja
Browse...
Options:
Compress the contents of the JAR file
Add directory entries
Overwrite existing files without warning
< Back
Next >
Finish
Cancel


JAR Export
JAR Packaging Options
Define the options for the JAR export.
Select options for handling problems:
Export class files with compile errors
Export class files with compile warnings
Create source folder structure
Build projects if not built automatically
Save the description of this JAR in the workspace
Description file:
Browse...
< Back
Next >
Finish
Cancel



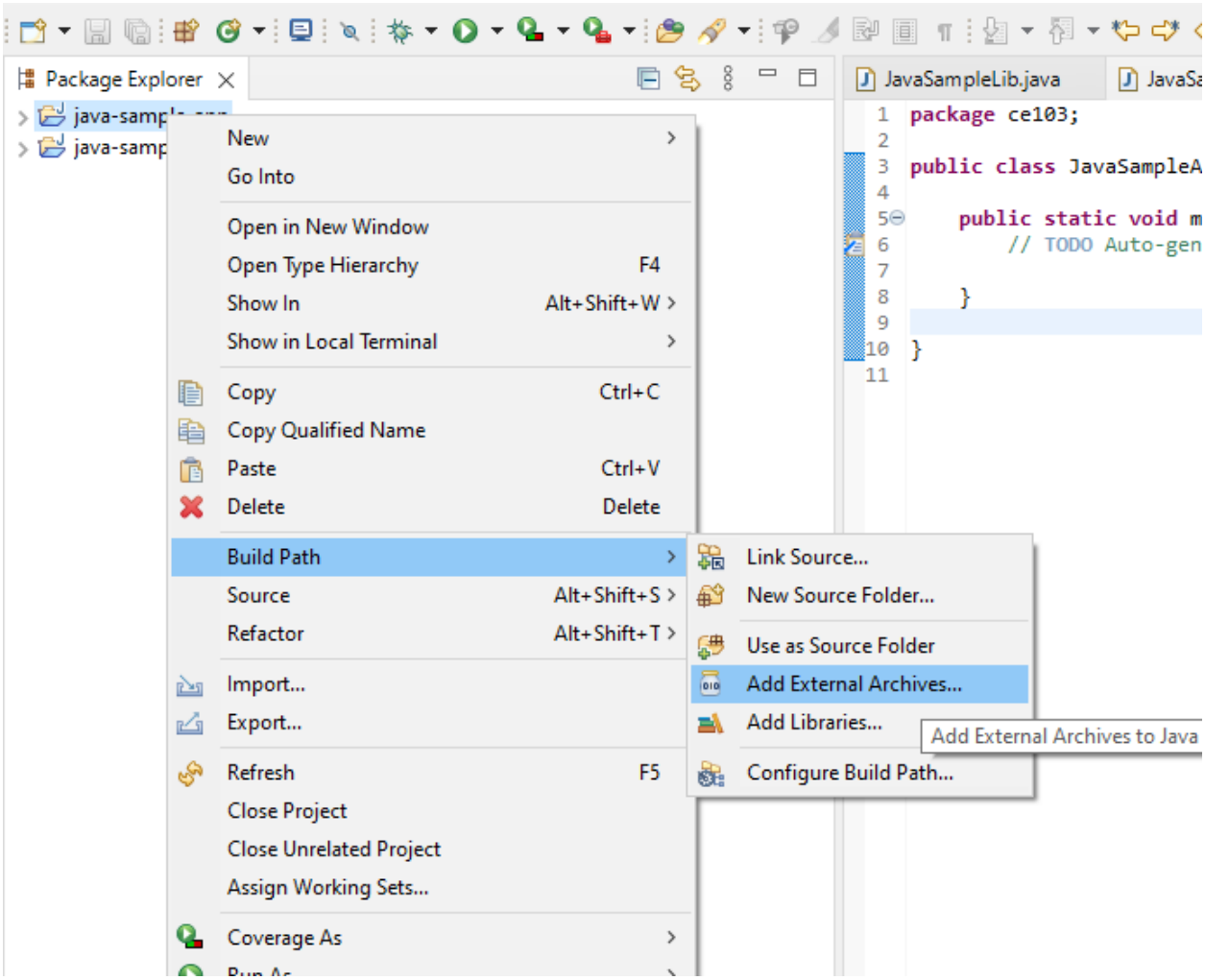
In the same export folder now we have JavaSampleLib.jar



---

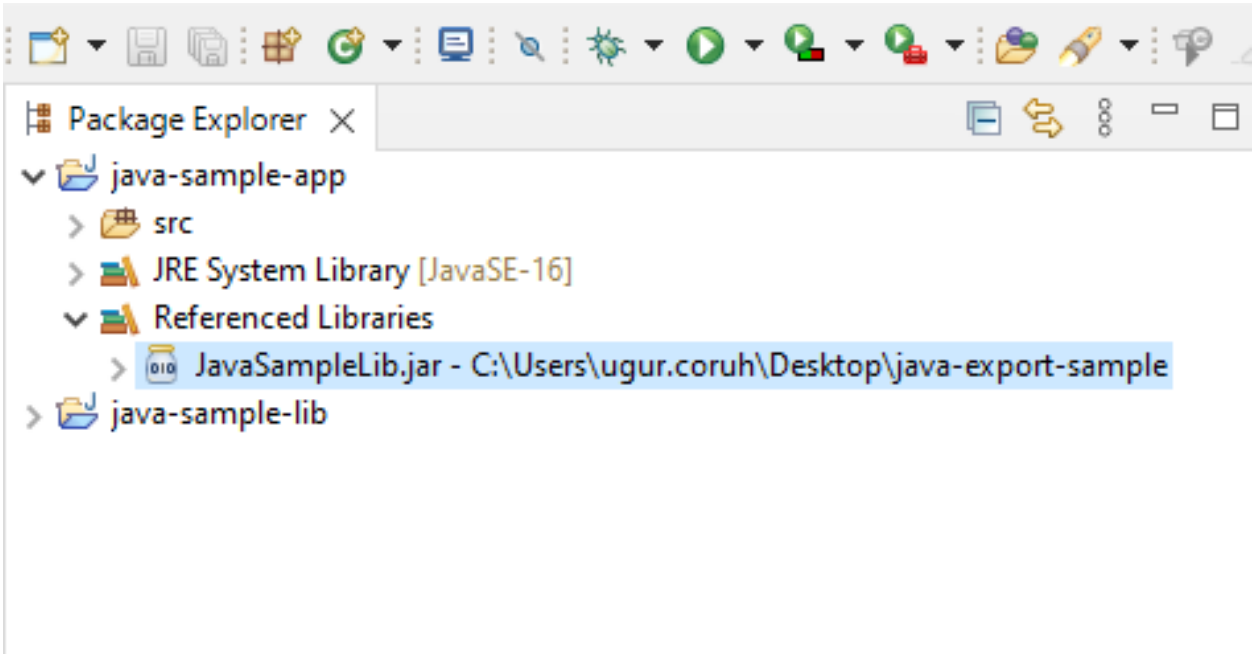
return back to java-sample-app and then add this jar file to our project

Build Path->Add External Archives



---

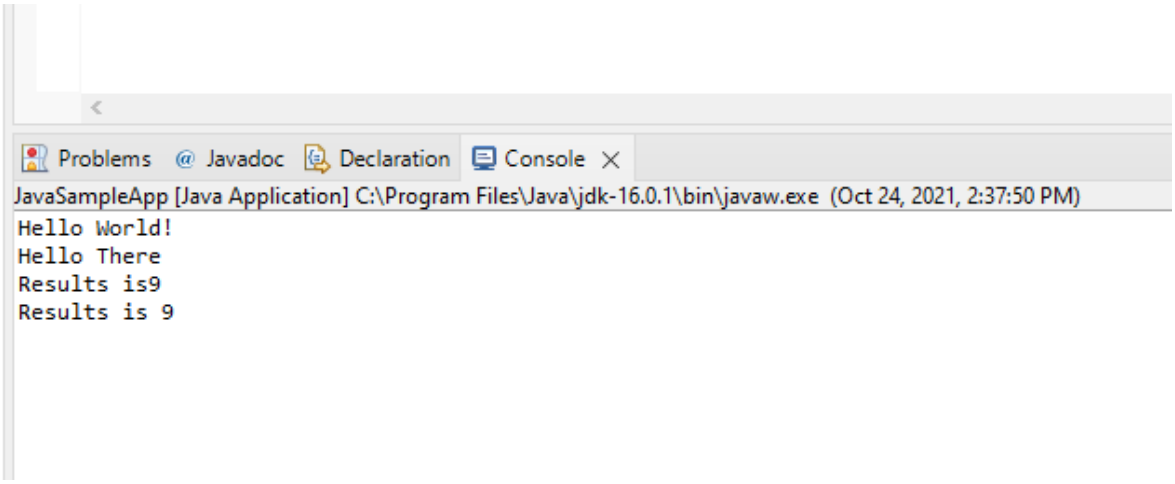
you will see its added to reference libraries



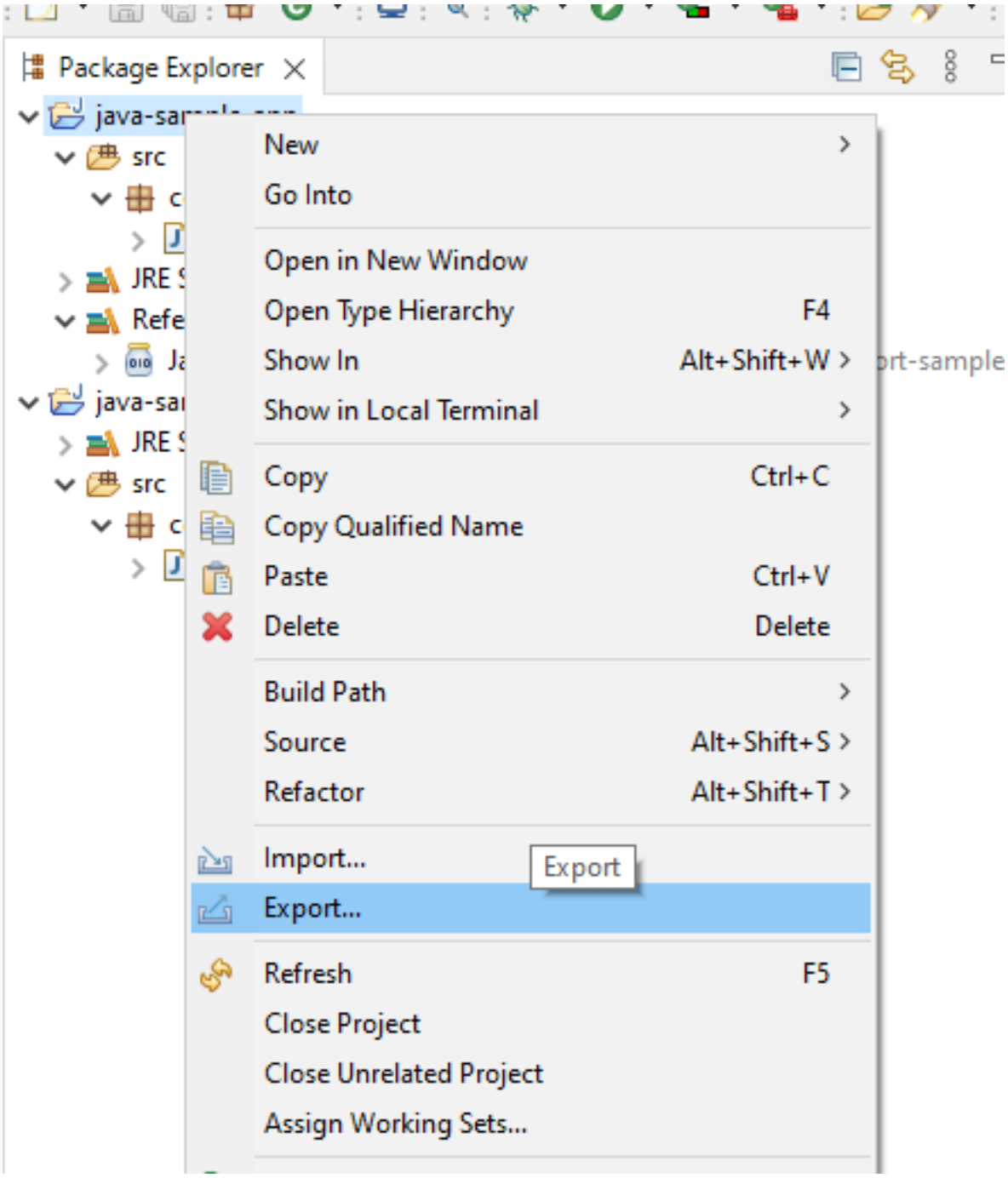
in our JavaSampleApp.java we can use the following source codes

```
package ce103;

import java.io.IOException;

public class JavaSampleApp {

    public static void main(String[] args) {
        // TODO Auto-generated method stub

        System.out.println("Hello World!");

        JavaSampleLib.sayHelloTo("Computer");
        int result = JavaSampleLib.sum(5, 4);
        System.out.println("Results is" + result);
        System.out.printf("Results is %d \n", result);


        try {
            System.in.read();
        } catch (IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }

    }

}
```

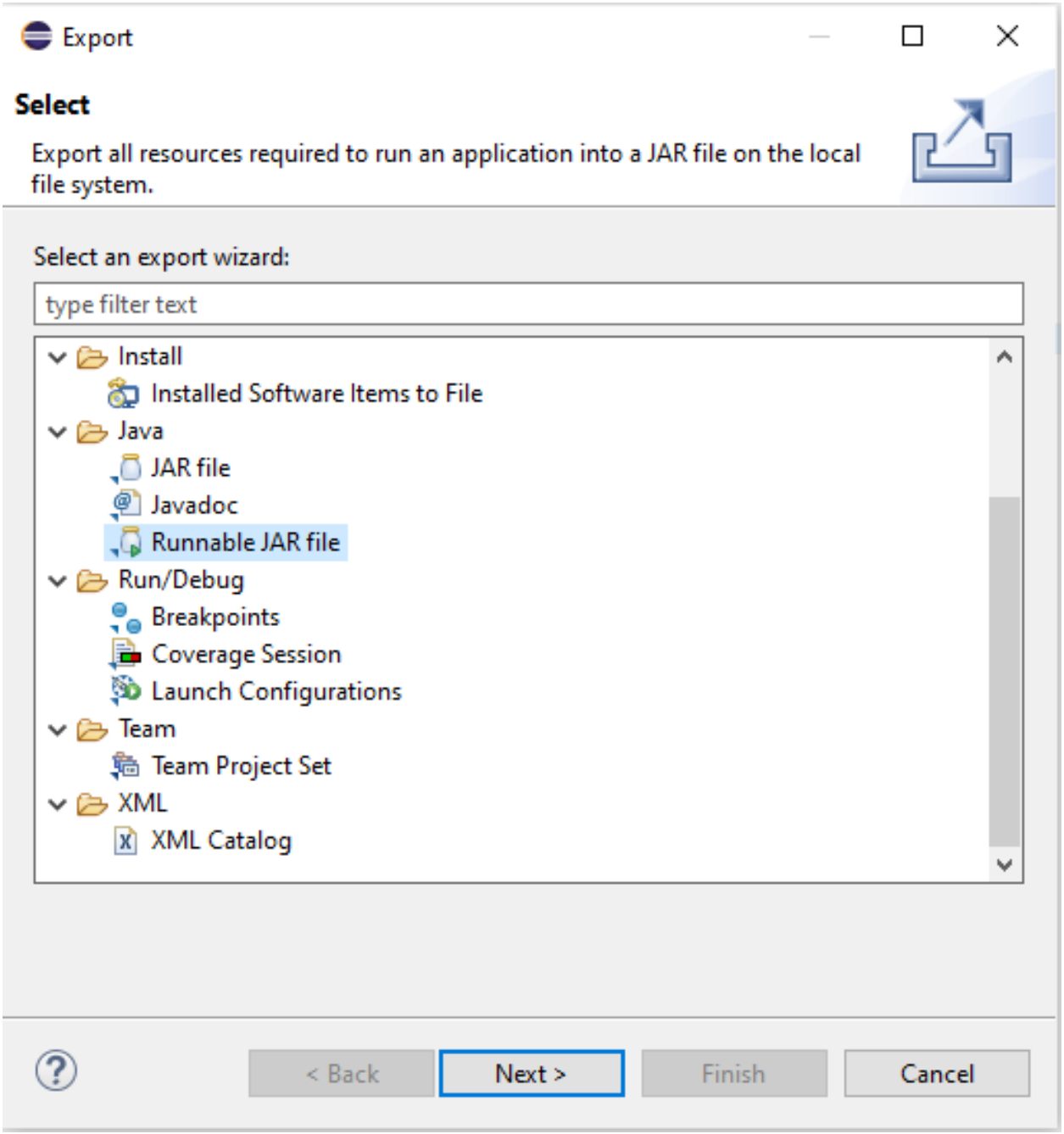
When we run application we will see similar output



Lets export this application with its dependent library

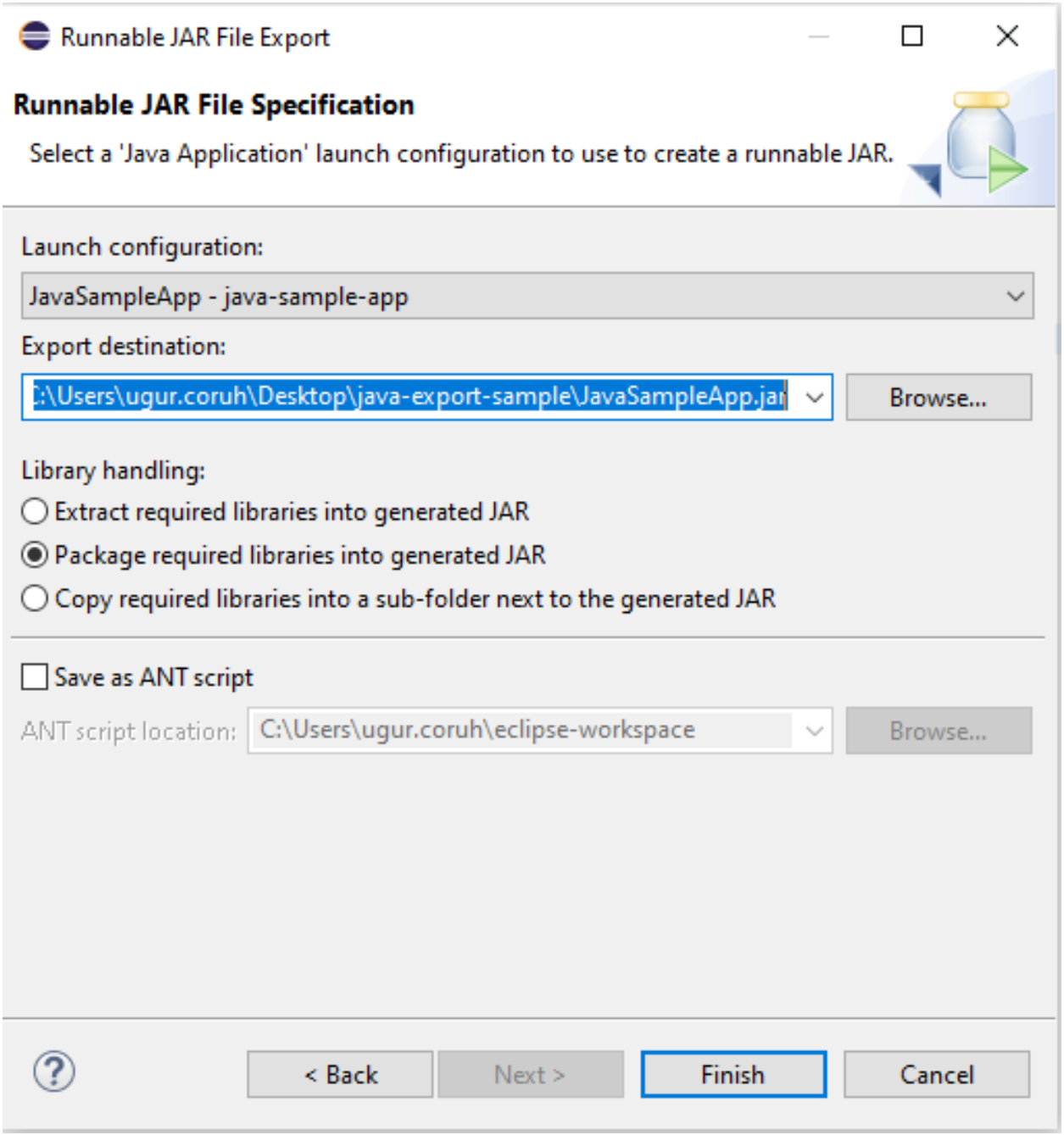


Select runnable jar



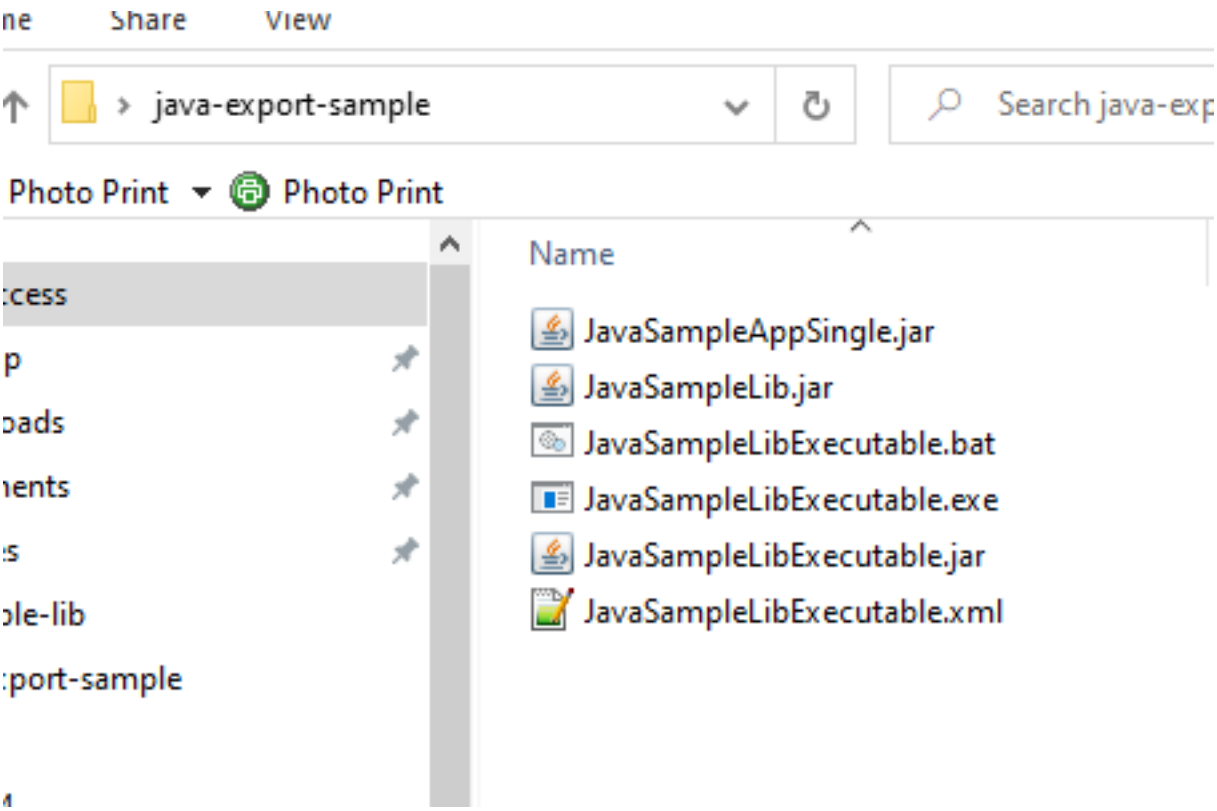
Set Launch configuration and Export destination

`C:\Users\ugur.coruh\Desktop\java-export-sample\JavaSampleAppSingle.jar`

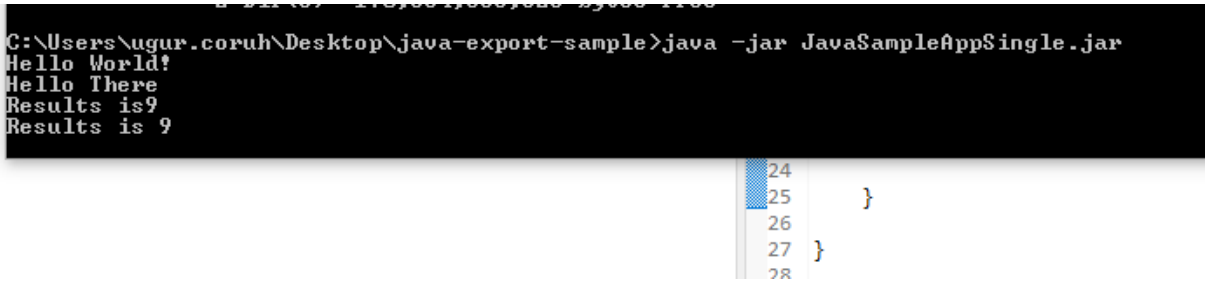


In this option we will have single jar file

In the export folder we do not see reference libraries



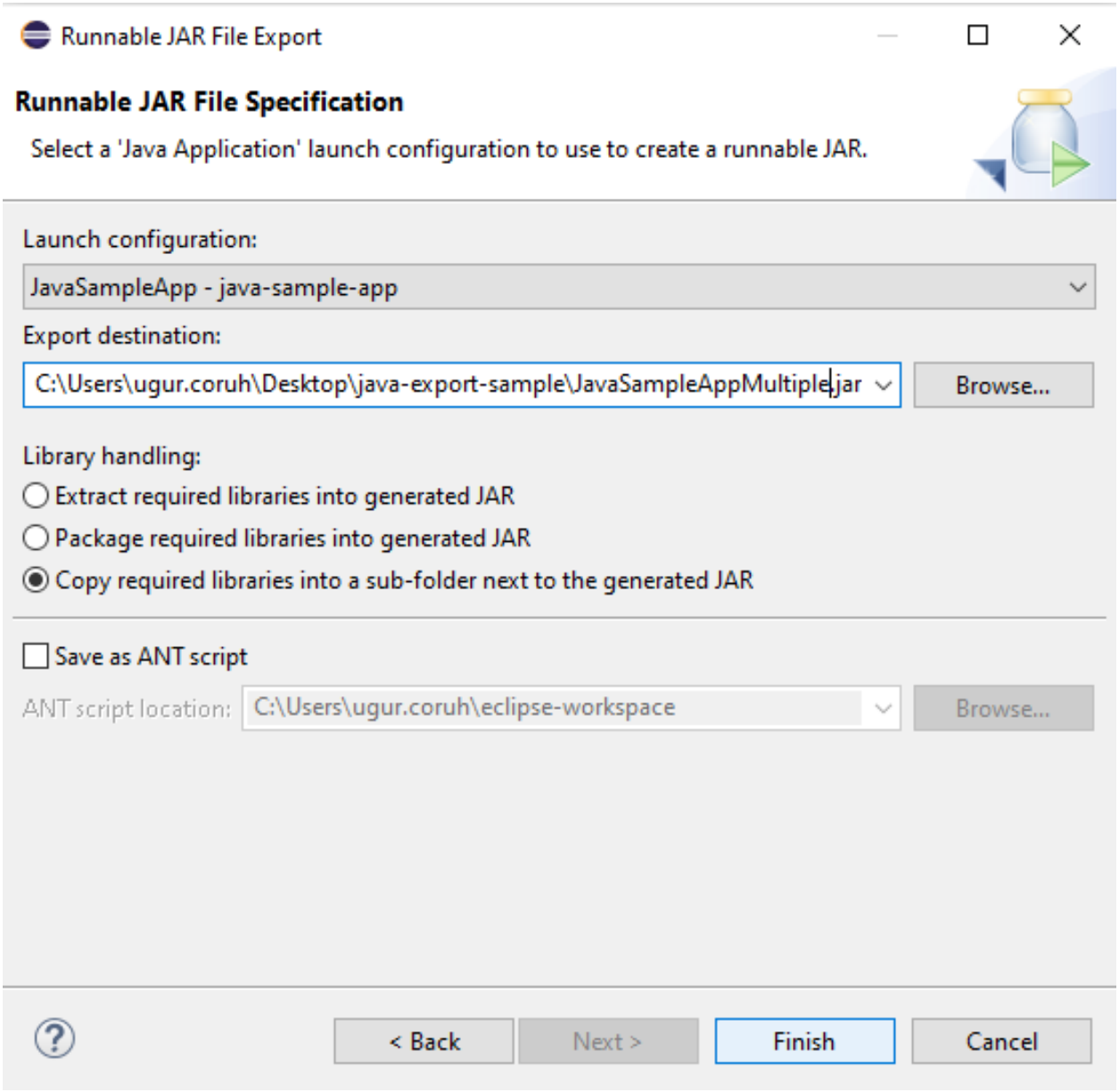
and we can run with command line



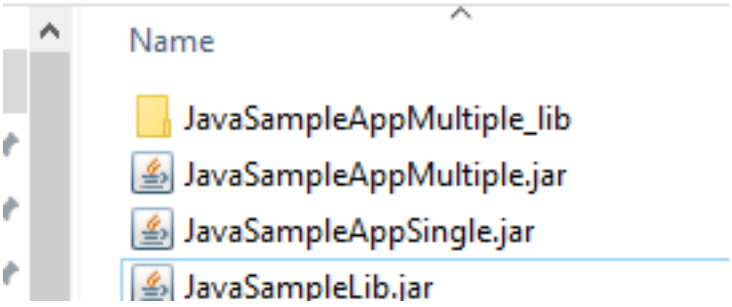
---

only change copy required libraries setting and then give a new name for new jar file and export

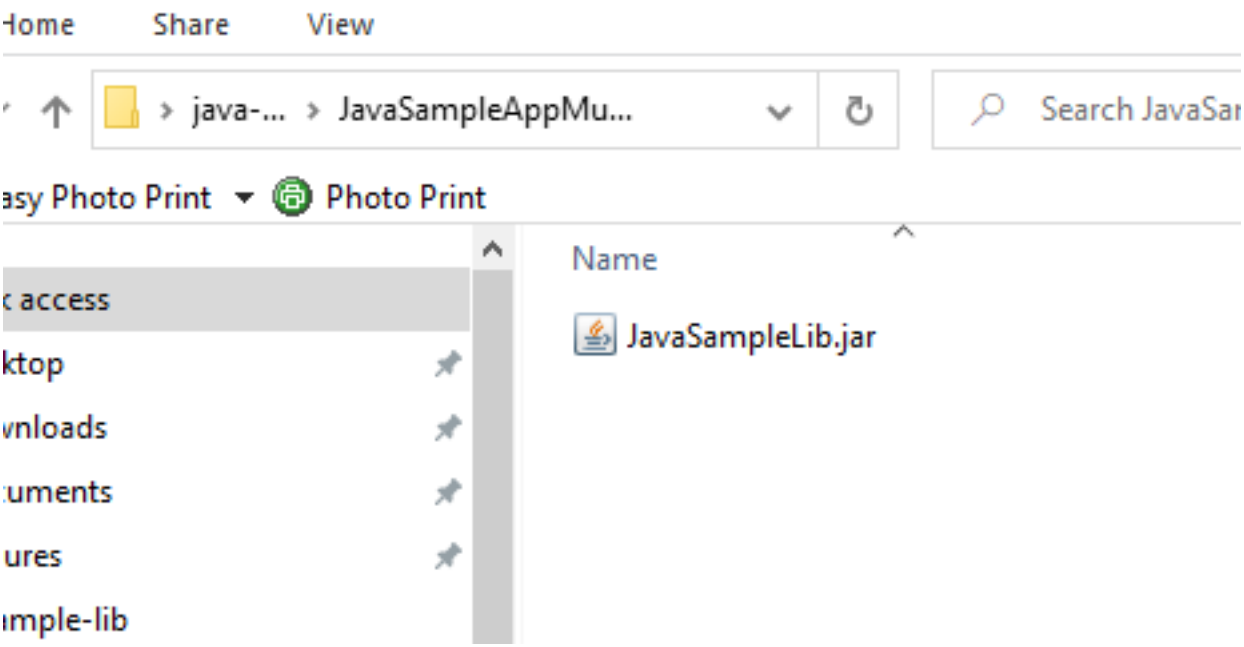
`C:\Users\ugur.coruh\Desktop\java-export-sample\JavaSampleAppMultiple.jar`



now we have a folder that contains our libraries referenced

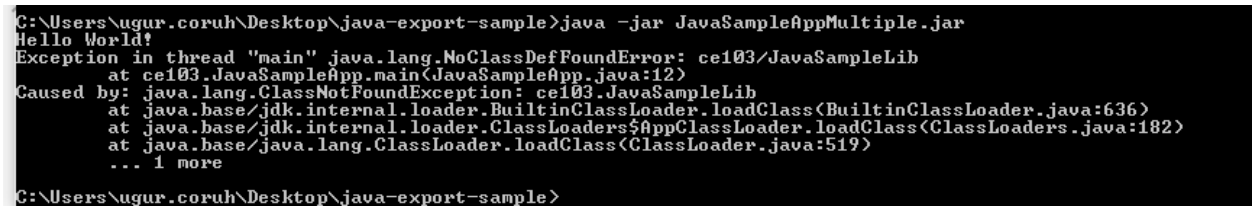


in this file we can find our library



if we test our application we will see it will work

if we delete JavaSampleLib.jar and then try running application we will get error



0.5 Program Testing

0.6 Unit Test Development

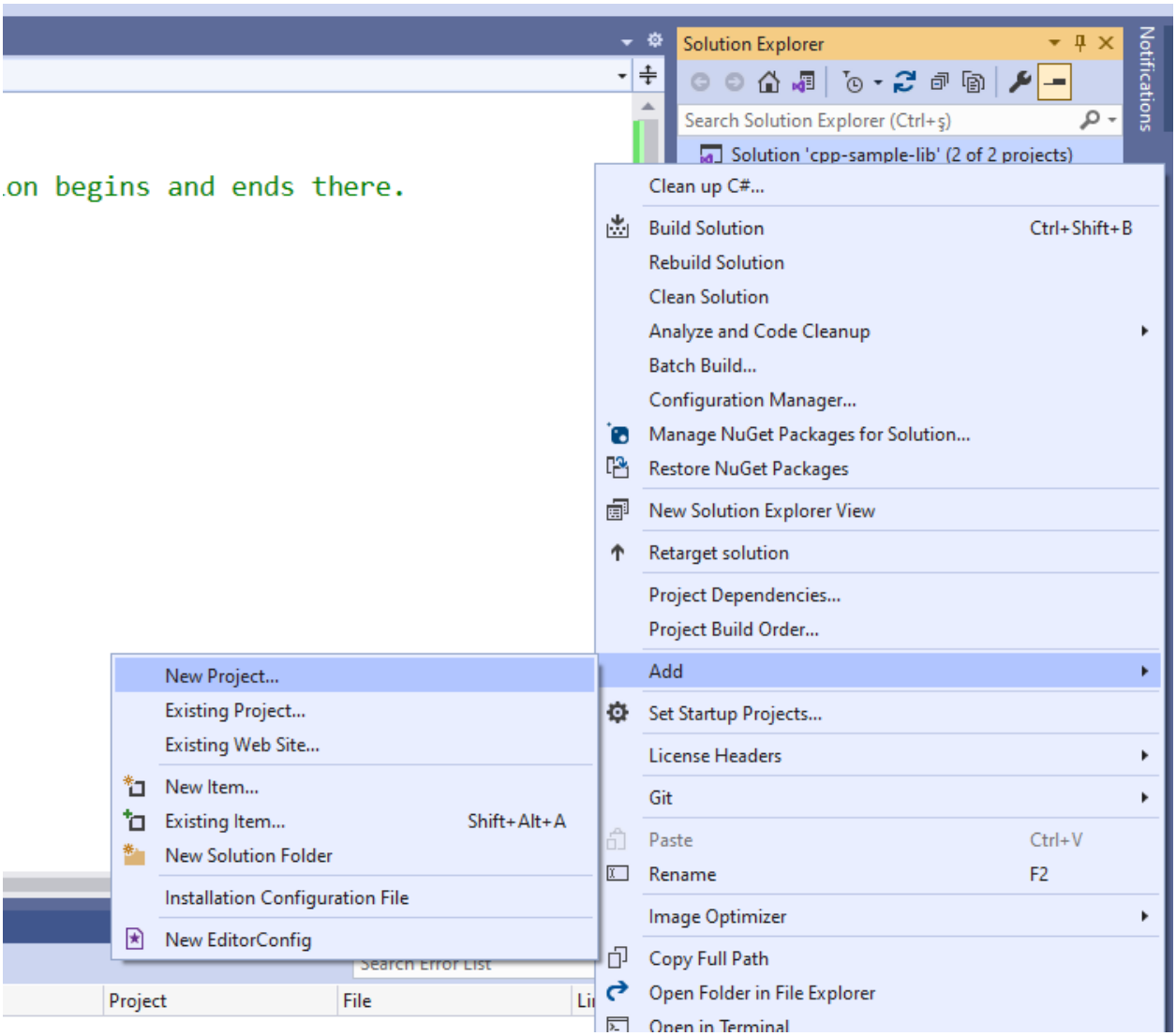
0.6.1 C Unit Tests

0.6.1.1 Visual Studio Community Edition

0.6.2 C++ Unit Tests

0.6.2.1 Visual Studio Community Edition C/C++ için birim testleri yazma - Visual Studio (Windows) | Microsoft Docs[12]

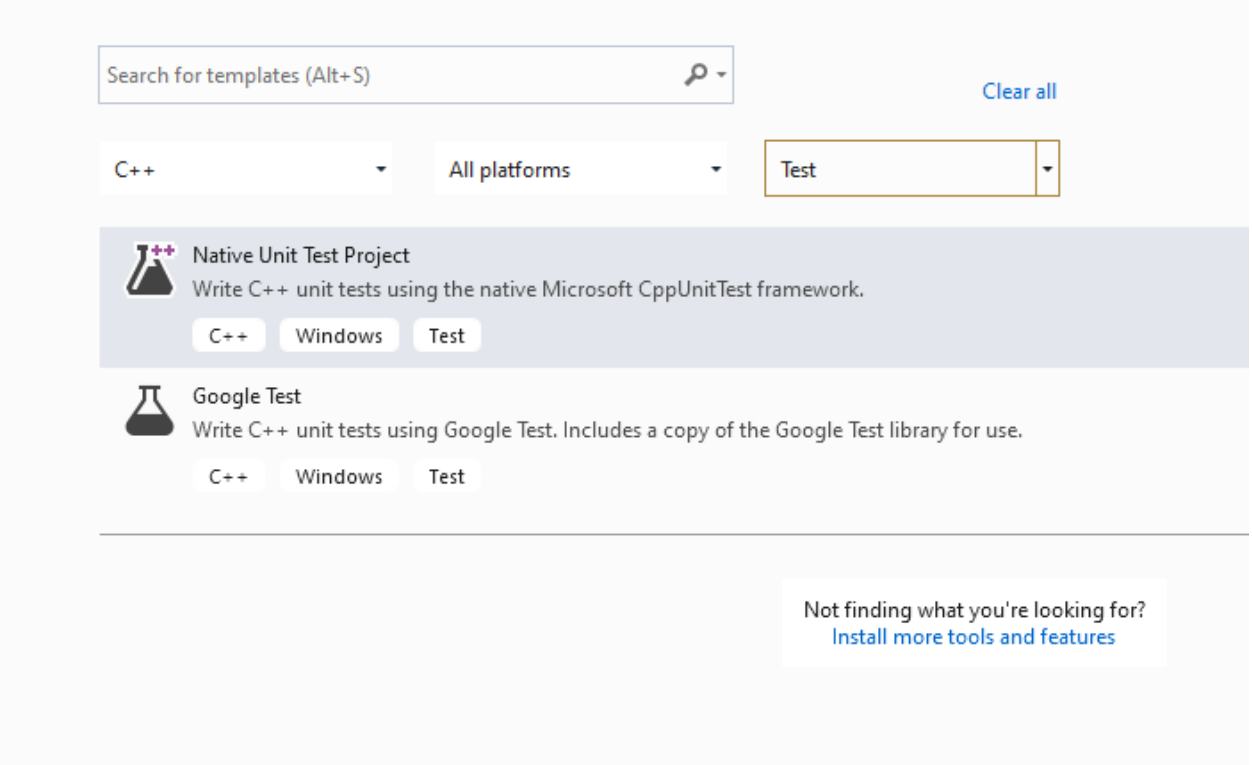
Use cpp-sample-lib project and add



---

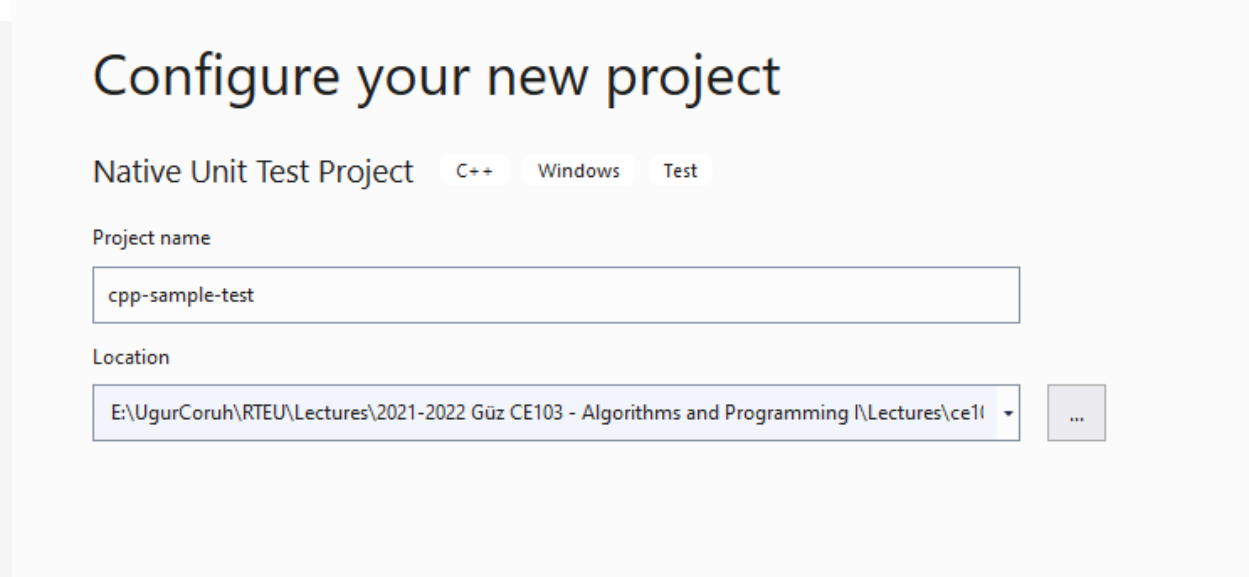
select Native Unit Test

[12] https://docs.microsoft.com/tr-tr/visualstudio/test/writing-unit-tests-for-c-cpp?view=vs-2019



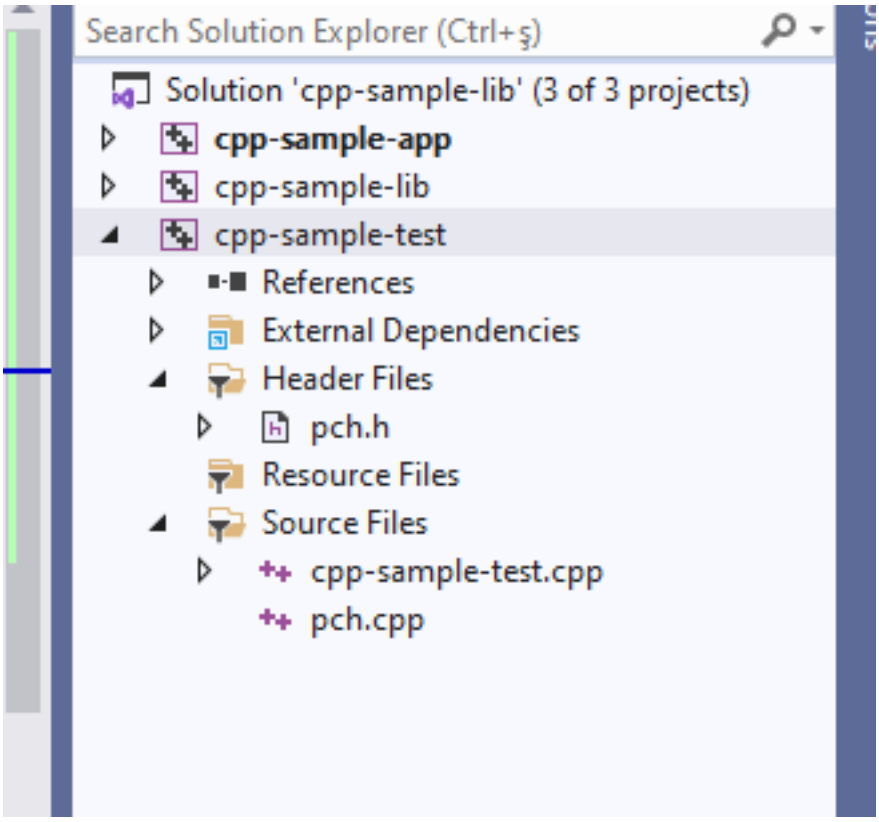
---

set project path and name

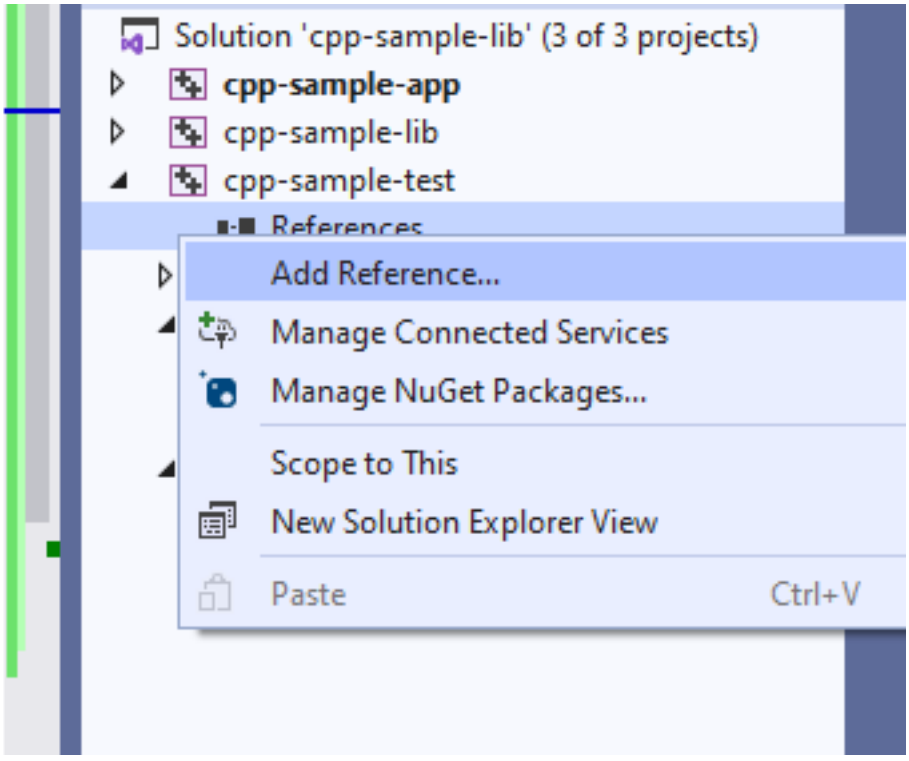


---

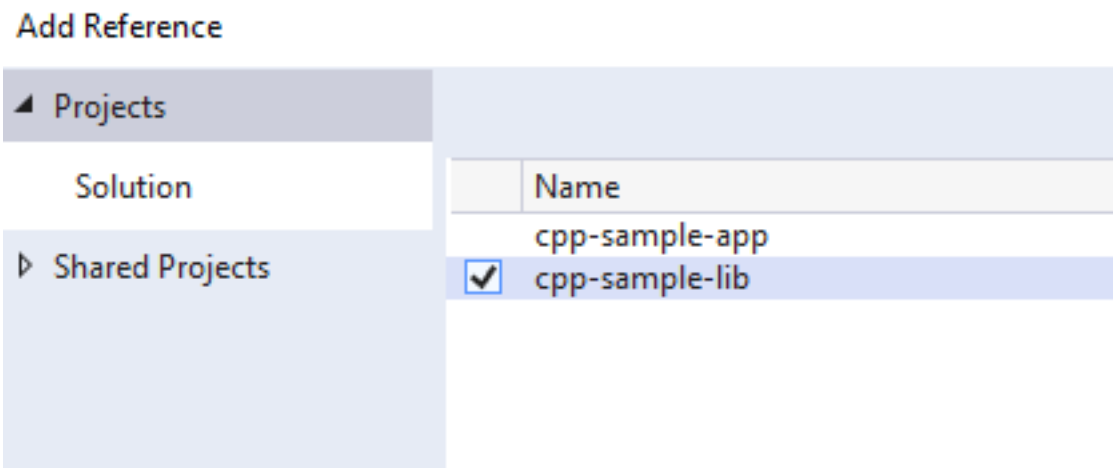
you will have cpp-sample-test project



add library project from references



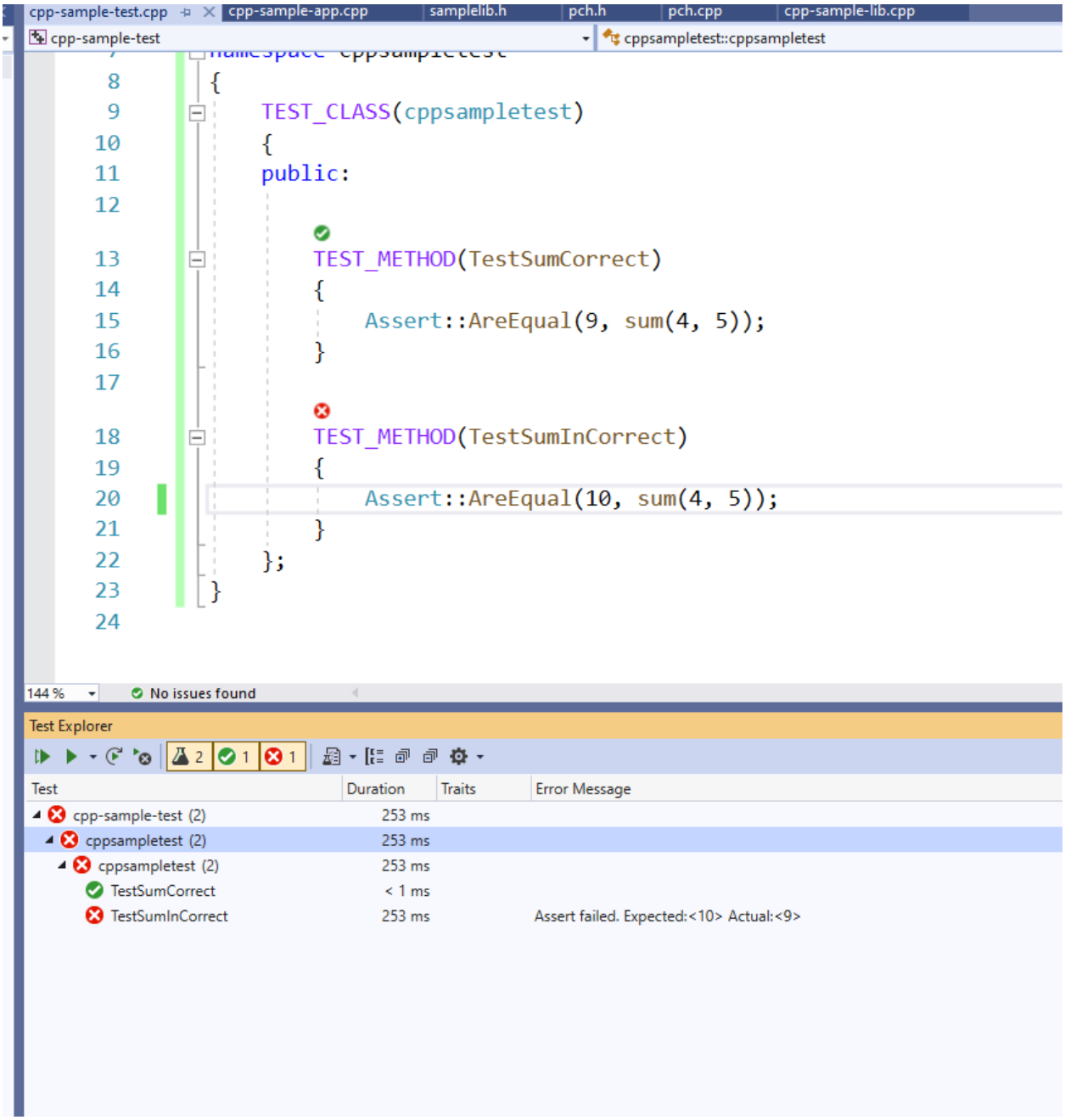
Add cpp-sample-lib to cpp-sample-test project



---

cpp-sample-test.cpp

```
#include "pch.h"
#include "CppUnitTest.h"
#include "..\cpp-sample-lib\samplelib.h"

using namespace Microsoft::VisualStudio::CppUnitTestFramework;

namespace cppsampletest
{
    TEST_CLASS(cppsampletest)
    {
    public:

        TEST_METHOD(TestSumCorrect)
        {
            Assert::AreEqual(9, sum(4, 5));
        }

        TEST_METHOD(TestSumInCorrect)
        {
            Assert::AreEqual(10, sum(4, 5));
        }
    };
}
```



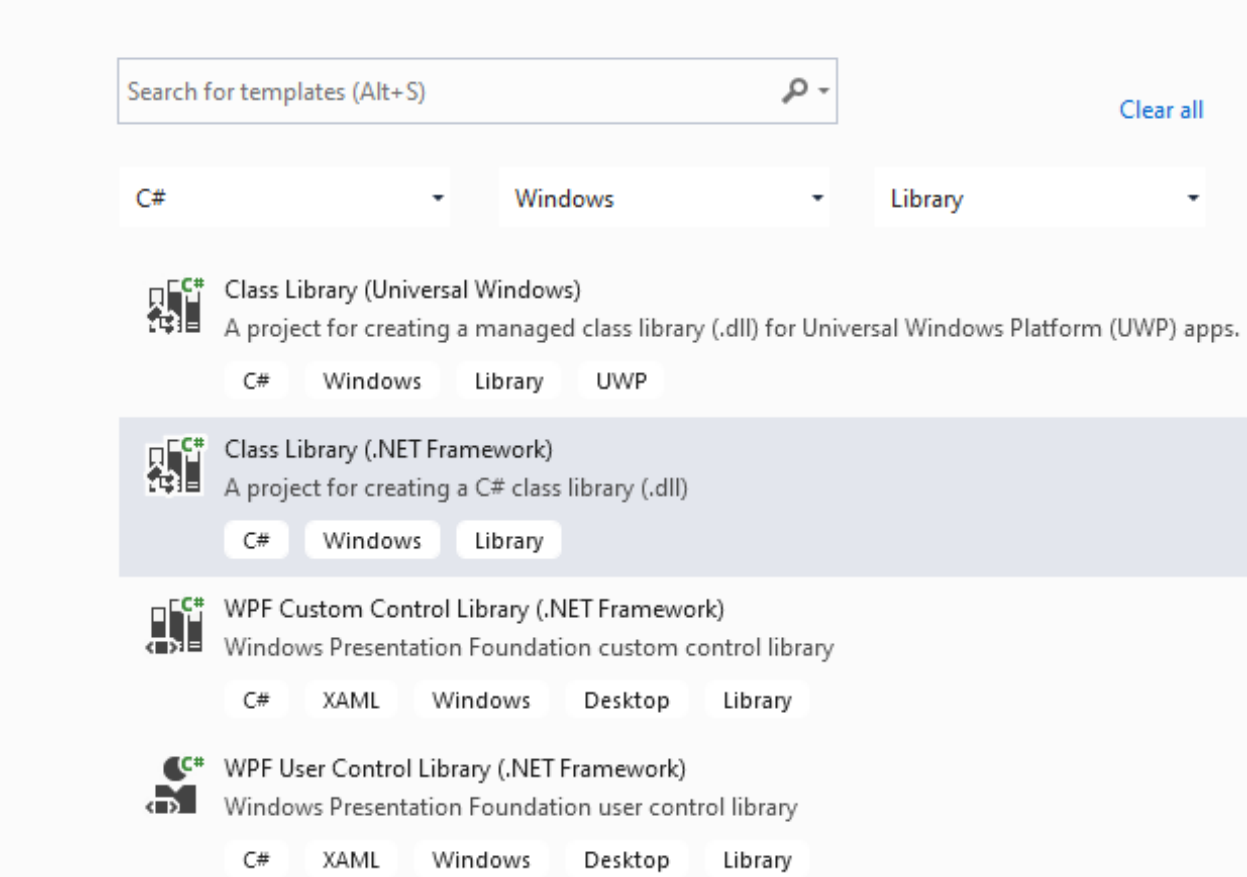
0.6.3 C# Unit Tests

0.6.4 Visual Studio Community Edition (MSTestV2+.Net)

Install extension fine code coverage

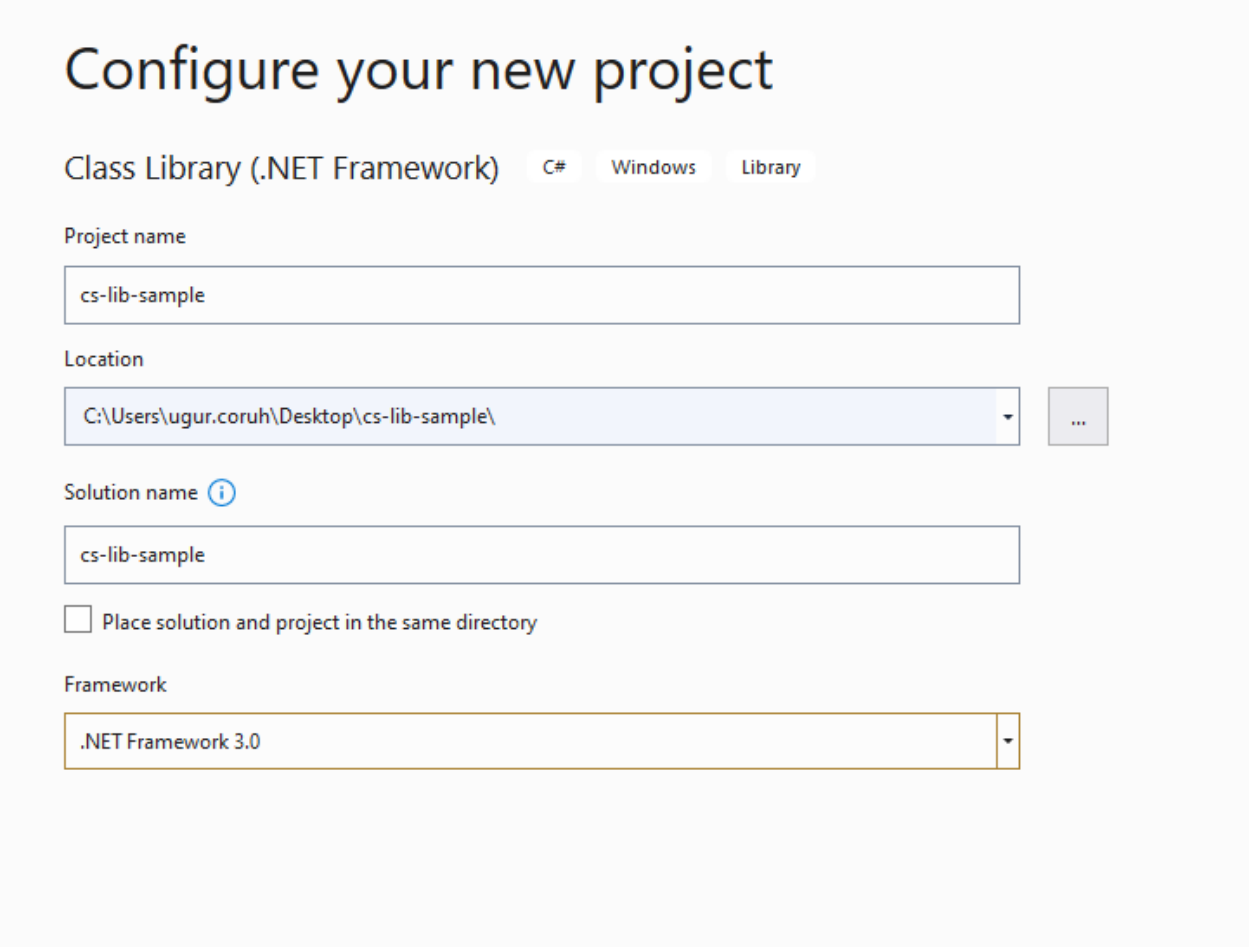
https://marketplace.visualstudio.com/items?itemName=FortuneNgwenya.FineCodeCoverage

Create a .Net Framework Library

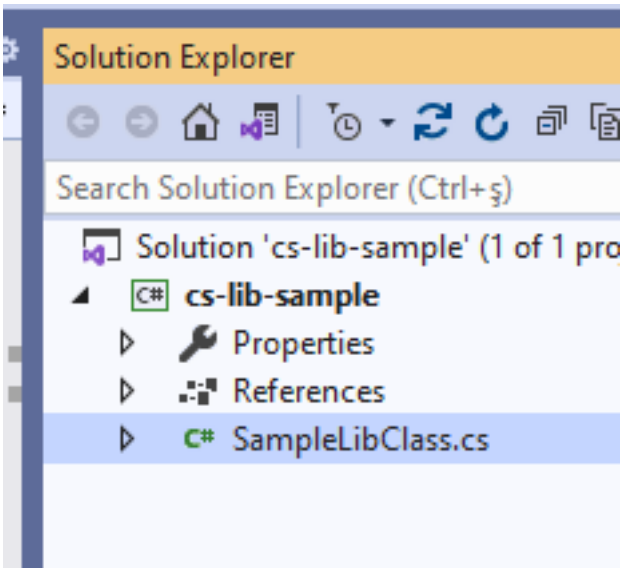


---

set project framework and path



Create library functions

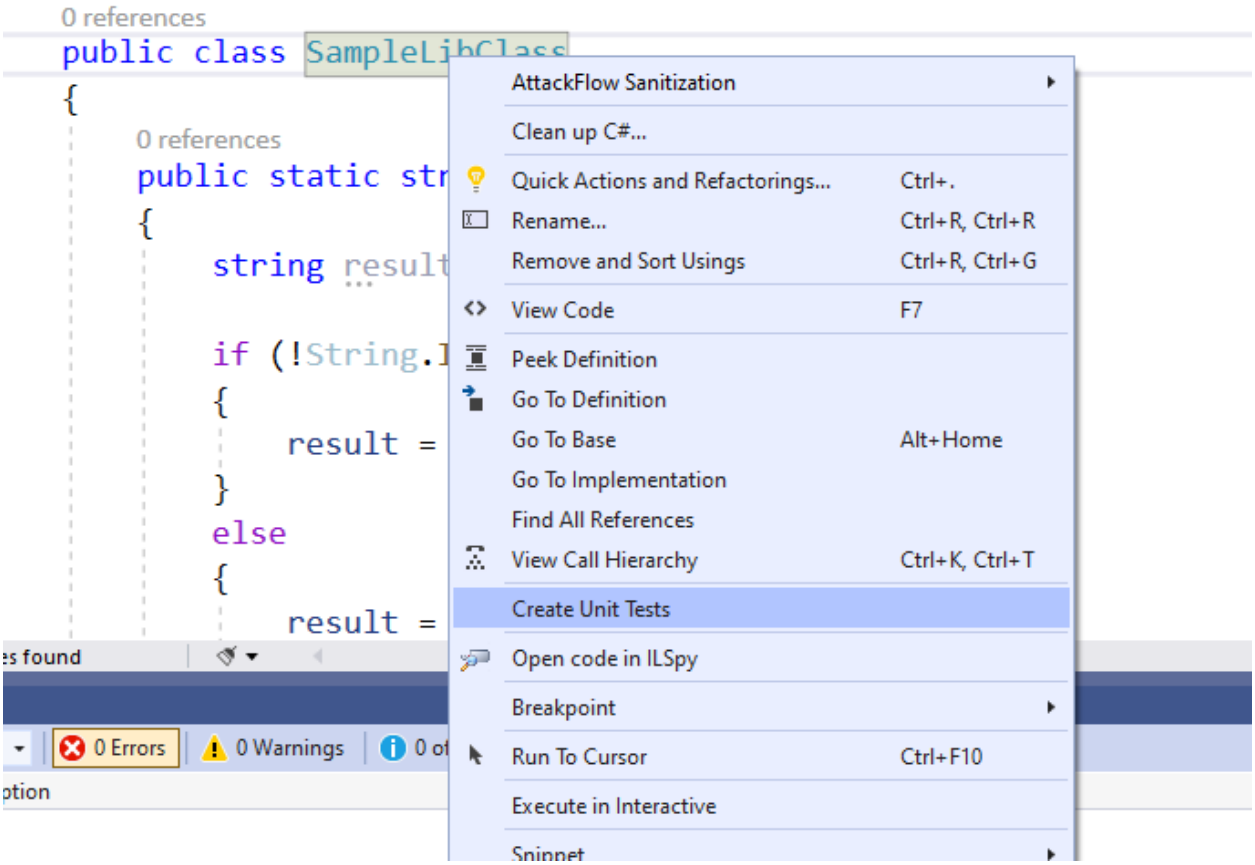


```
using System;
using System.Collections.Generic;
using System.Text;
```

```
namespace cs_lib_sample
{
    public class SampleLibClass
    {
        public static string sayHelloTo(string name)
        {
            string result = String.Empty;

            if (!String.IsNullOrEmpty(name))
            {
                result = "Hello " + name;
            }
            else
            {
                result = "Hello There";
            }

            Console.WriteLine(result);

            return result;
        }

        public static int sum(int a, int b)
        {
            int c = 0;
            c = a + b;
            return c;
        }

        public int multiply(int a, int b)
        {
            return a * b;
        }
    }
}
```

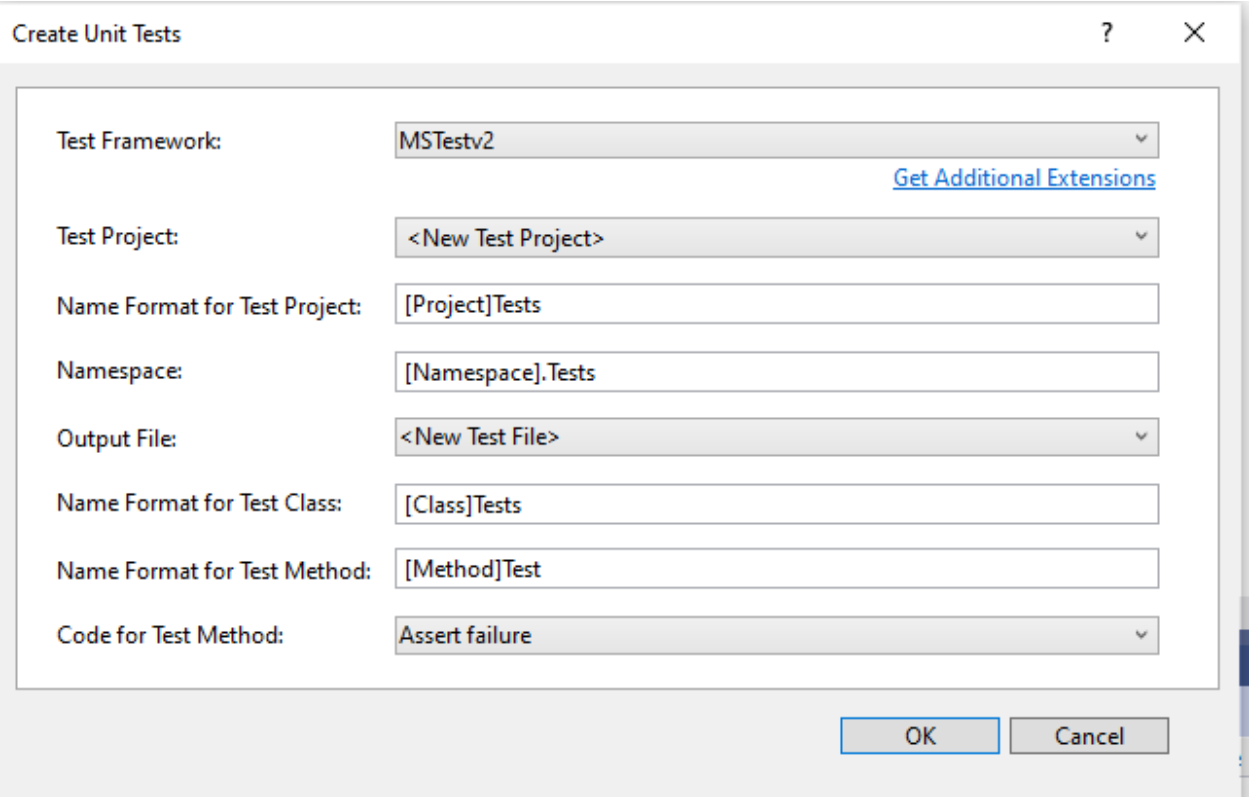
---

right click and then create unit test project


0 references
public class SampleLibClass
{
0 references
public static str
{
string result
if (!String.I
{
result =
}
else
{
result =
AttackFlow Sanitization
Clean up C#...
Quick Actions and Refactorings... Ctrl+.
Rename... Ctrl+R, Ctrl+R
Remove and Sort Usings Ctrl+R, Ctrl+G
View Code F7
Peek Definition
Go To Definition
Go To Base Alt+Home
Go To Implementation
Find All References
View Call Hierarchy Ctrl+K, Ctrl+T
Create Unit Tests
Open code in ILSpy
Breakpoint
Run To Cursor Ctrl+F10
Execute in Interactive
Snippet
0 Errors
0 Warnings

---

press OK



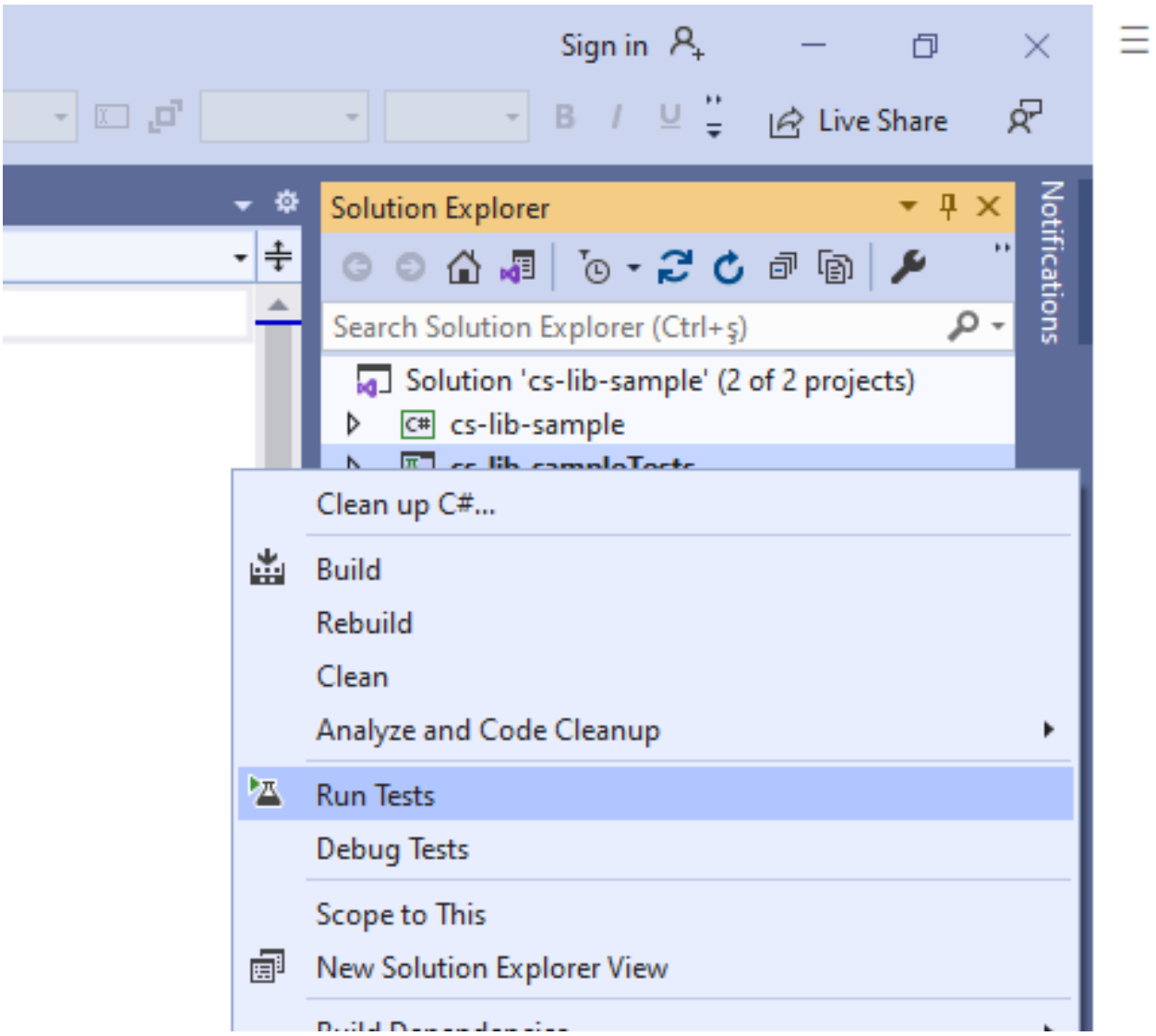
---

enter test code

```
using Microsoft.VisualStudio.TestTools.UnitTesting;
using cs_lib_sample;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace cs_lib_sample.Tests
{
    [TestClass()]
    public class SampleLibClassTests
    {

        [TestMethod()]
        public void testSayHelloTo()
        {


            Assert.AreEqual("Hello Computer", SampleLibClass.sayHelloTo("Computer"), "Regular say hello
        }
        [TestMethod()]
        public void testSayHelloToWrong()
        {
            Assert.AreEqual("Hello All", SampleLibClass.sayHelloTo("Computer"), "Regular say hello won'
        }
```

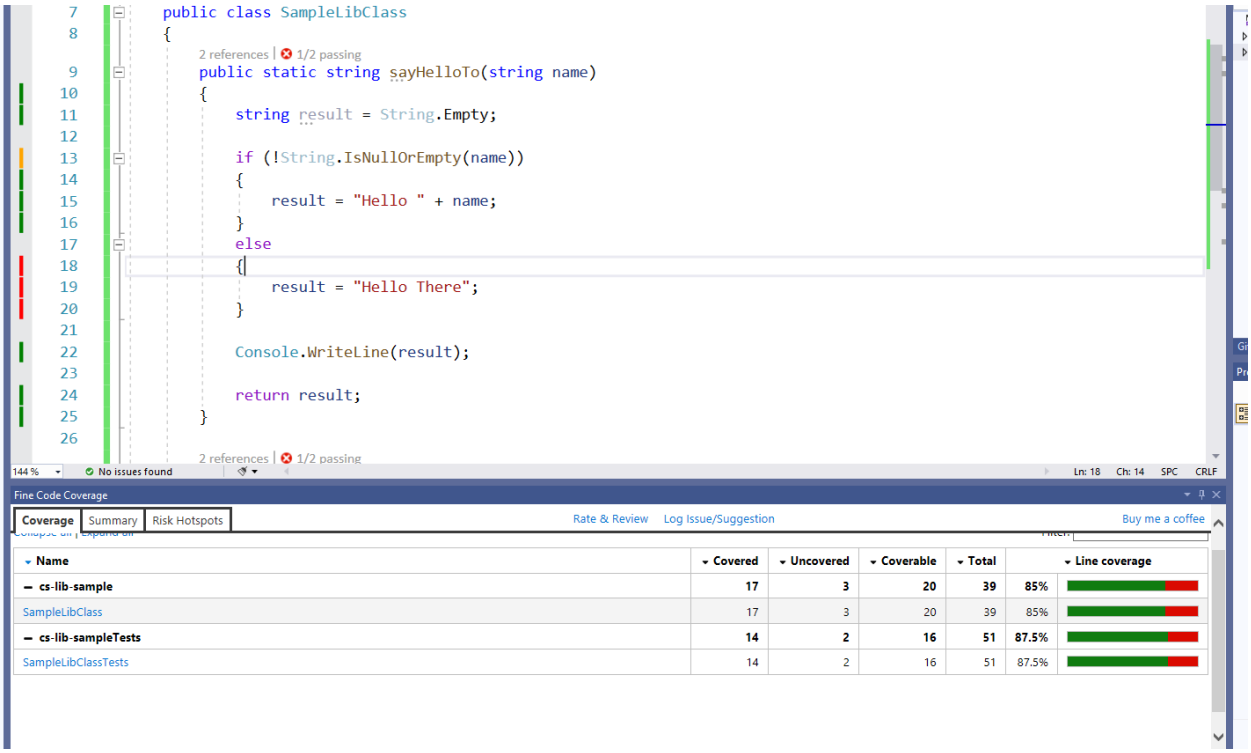
```
        [TestMethod()]
        public void testSumCorrect()
        {
            Assert.AreEqual(9, SampleLibClass.sum(4, 5), "Regular sum should work");
        }

        [TestMethod()]
        public void testSumWrong()
        {
            Assert.AreEqual(10, SampleLibClass.sum(4, 5), "Regular sum shouldn't work");
        }

        [TestMethod()]
        public void testMultiply()
        {
            SampleLibClass sampleLib = new SampleLibClass();

            Assert.AreEqual(20, sampleLib.multiply(4, 5), "Regular multiplication should work");
        }


    }
}
```

Run tests



---

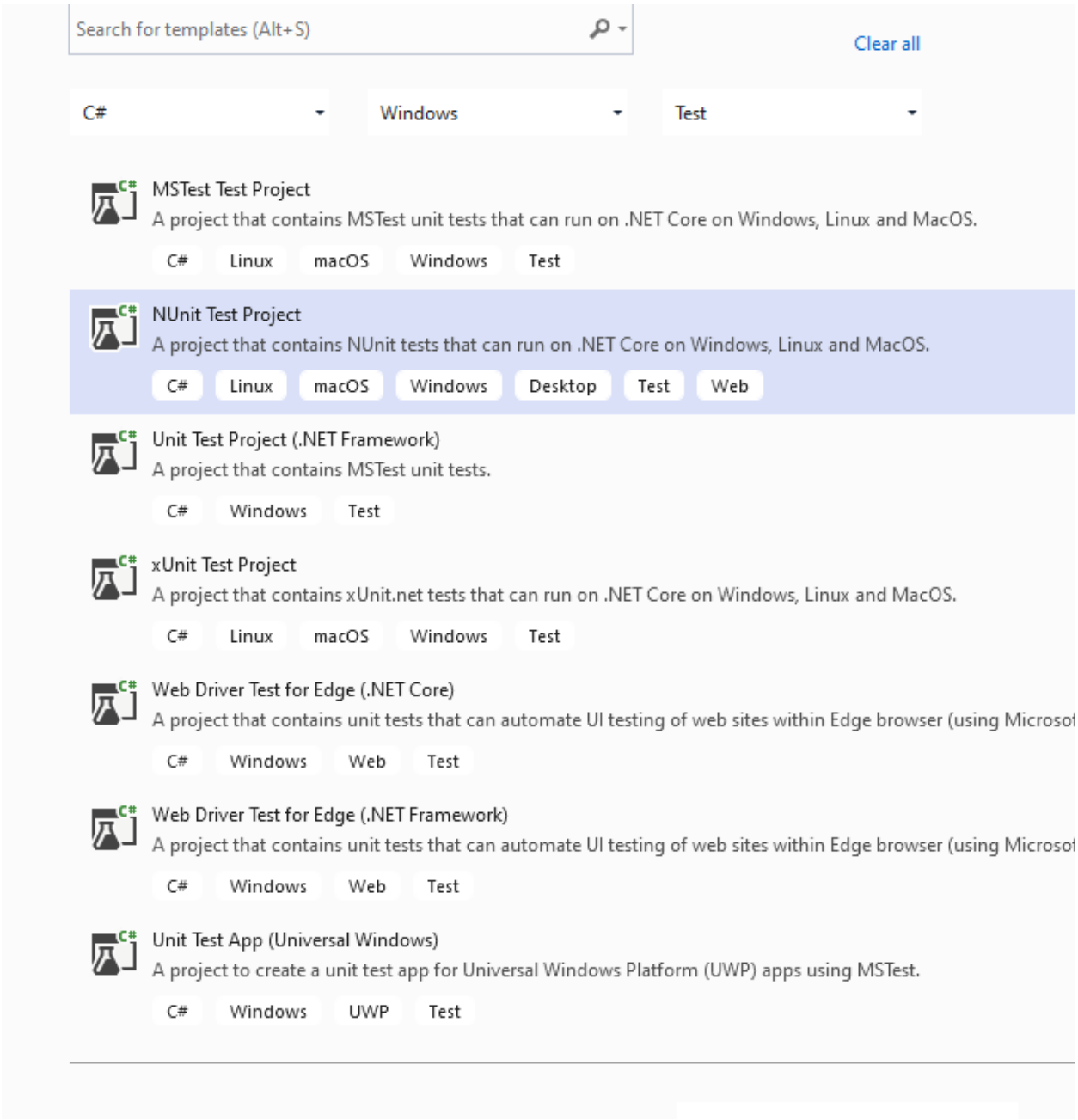
you will code coverage and entered or passed branches

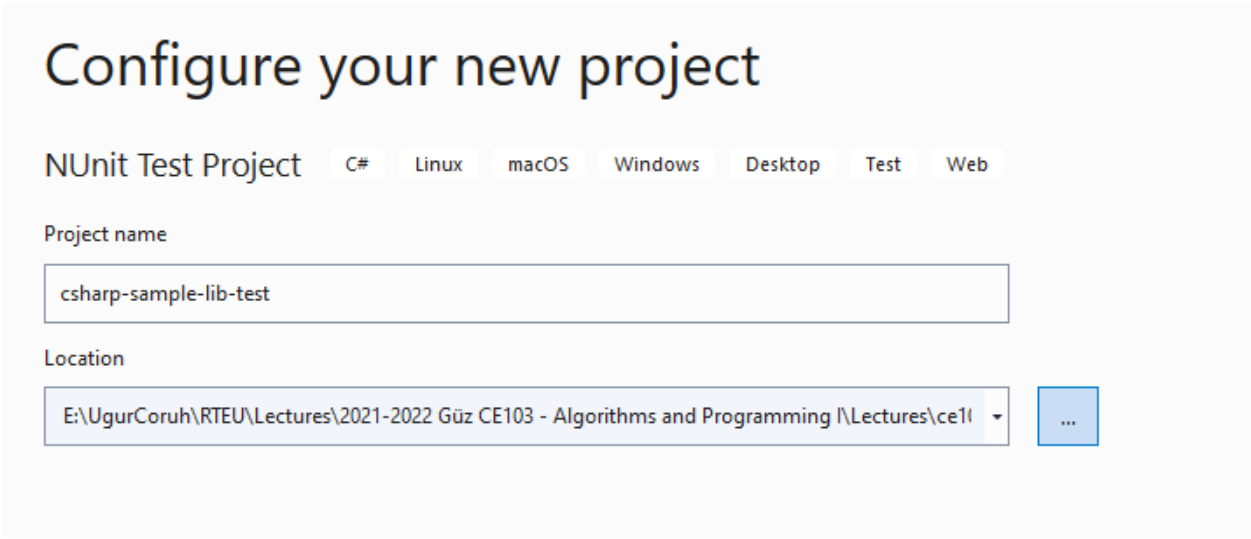


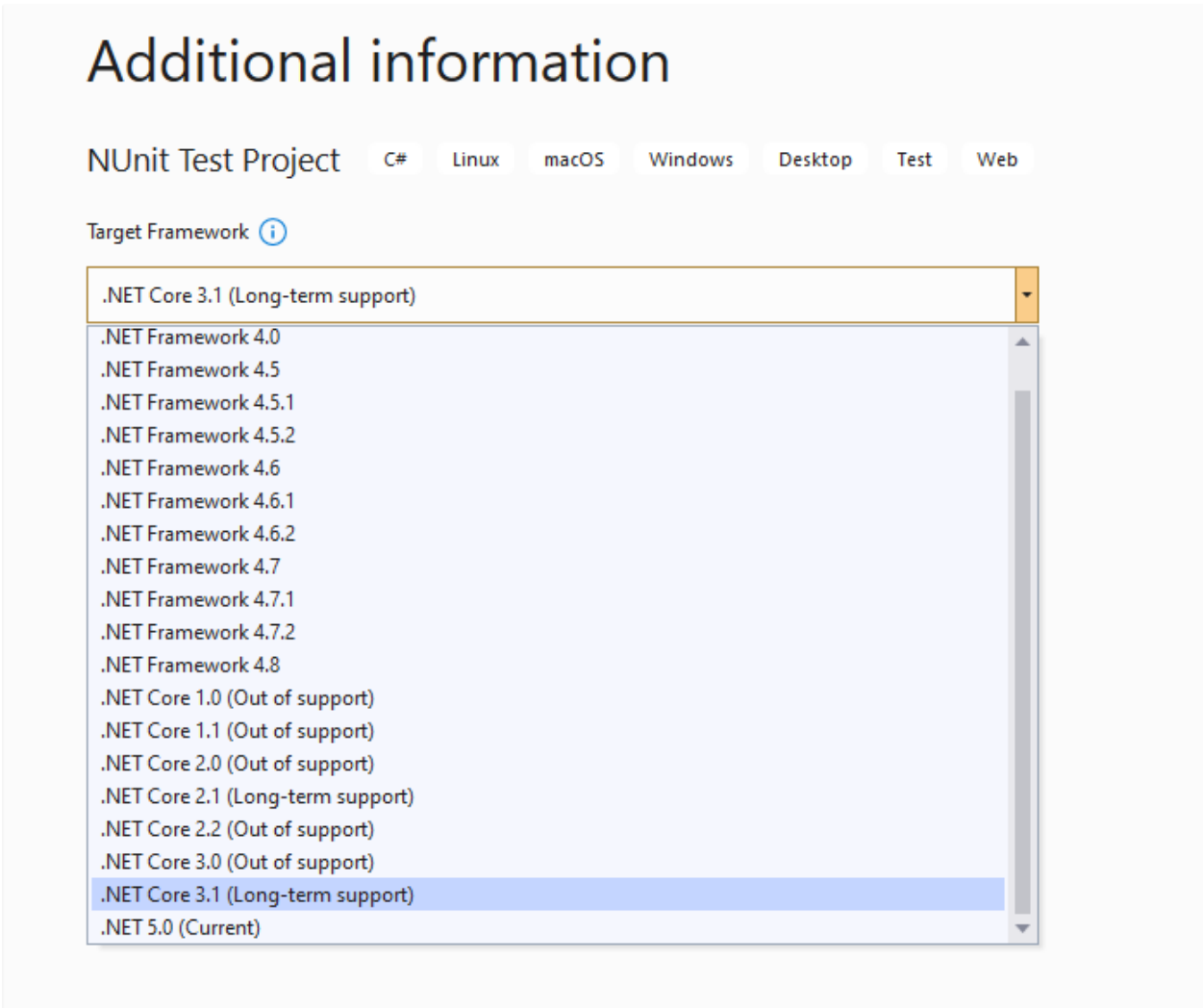
0.6.4.1 Visual Studio Community Edition (NUnit+.NETCore)

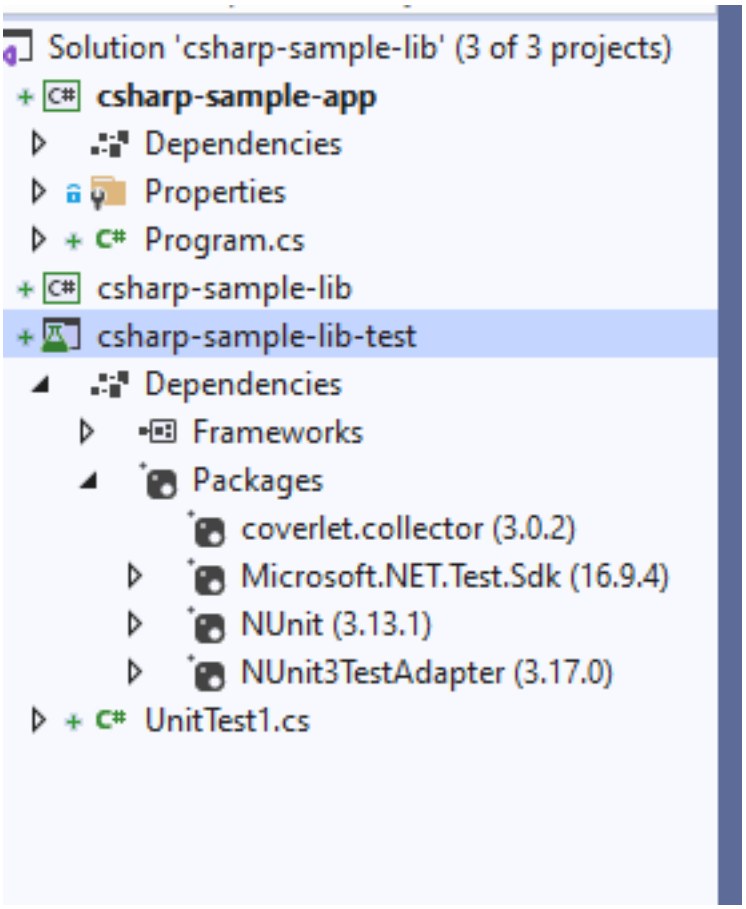
use cshar-sample-lib for this example

create and add a unit test project to solution

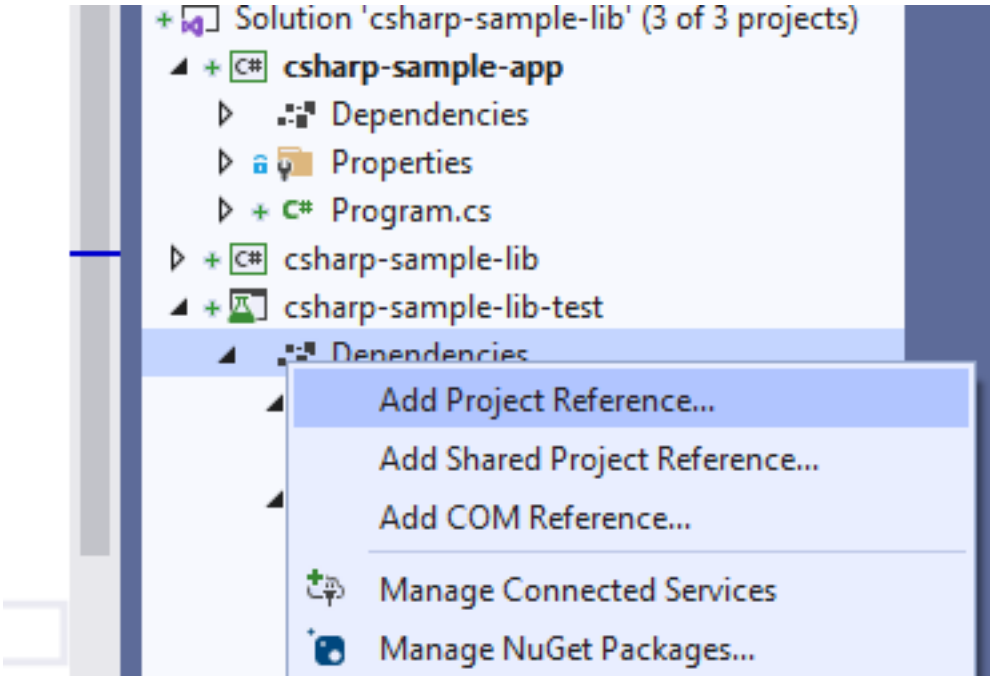

Search for templates (Alt+S)
Clear all
C#
Windows
Test
MSTest Test Project
A project that contains MSTest unit tests that can run on .NET Core on Windows, Linux and MacOS.
C# Linux macOS Windows Test
NUnit Test Project
A project that contains NUnit tests that can run on .NET Core on Windows, Linux and MacOS.
C# Linux macOS Windows Desktop Test Web
Unit Test Project (.NET Framework)
A project that contains MSTest unit tests.
C# Windows Test
xUnit Test Project
A project that contains xUnit.net tests that can run on .NET Core on Windows, Linux and MacOS.
C# Linux macOS Windows Test
Web Driver Test for Edge (.NET Core)
A project that contains unit tests that can automate UI testing of web sites within Edge browser (using Microsof
C# Windows Web Test
Web Driver Test for Edge (.NET Framework)
A project that contains unit tests that can automate UI testing of web sites within Edge browser (using Microsof
C# Windows Web Test
Unit Test App (Universal Windows)
A project to create a unit test app for Universal Windows Platform (UWP) apps using MSTest.
C# Windows UWP Test

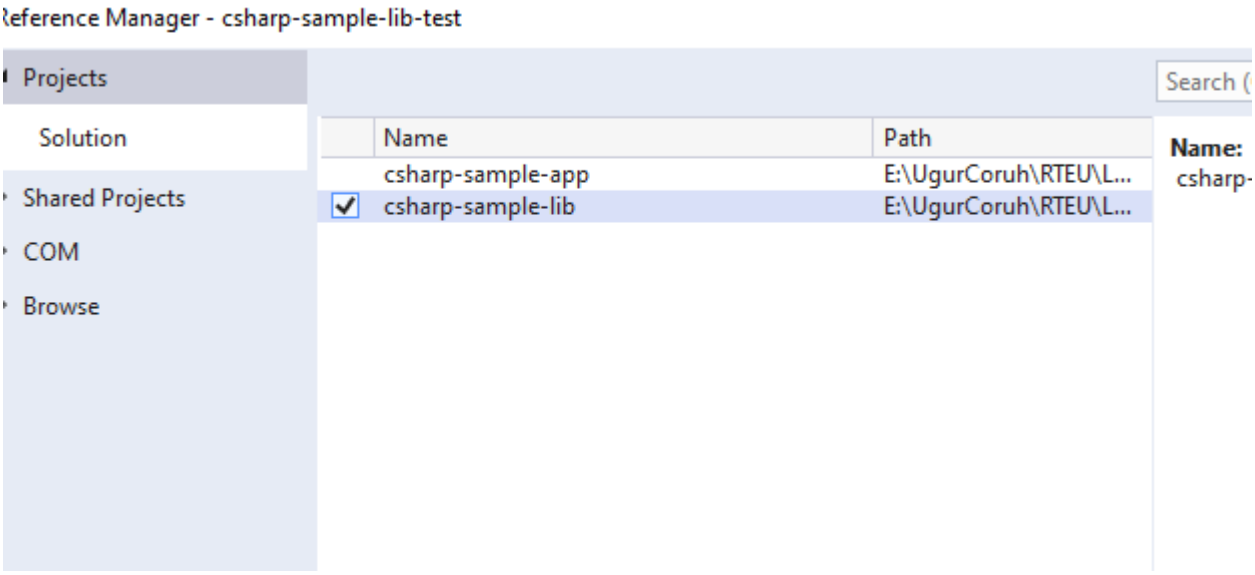

Configure your new project
NUnit Test Project
C#
Linux
macOS
Windows
Desktop
Test
Web
Project name
csharp-sample-lib-test
Location
E:\UgurCoruh\RTEU\Lectures\2021-2022 Güz CE103 - Algorithms and Programming I\Lectures\ce1
...


Additional information
NUnit Test Project
C#
Linux
macOS
Windows
Desktop
Test
Web
Target Framework
.NET Core 3.1 (Long-term support)
.NET Framework 4.0
.NET Framework 4.5
.NET Framework 4.5.1
.NET Framework 4.5.2
.NET Framework 4.6
.NET Framework 4.6.1
.NET Framework 4.6.2
.NET Framework 4.7
.NET Framework 4.7.1
.NET Framework 4.7.2
.NET Framework 4.8
.NET Core 1.0 (Out of support)
.NET Core 1.1 (Out of support)
.NET Core 2.0 (Out of support)
.NET Core 2.1 (Long-term support)
.NET Core 2.2 (Out of support)
.NET Core 3.0 (Out of support)
.NET Core 3.1 (Long-term support)
.NET 5.0 (Current)



Add project reference





SampleLibraryTestClasss in NUnit Project

```
using csharp_sample_lib;
using NUnit.Framework;

namespace csharp_sample_lib_test
{
    public class SampleLibraryTestClass
    {
        sampleLibClass sampleLib;

        [SetUp]
        public void Setup()
        {
            sampleLib = new sampleLibClass();
        }

        [Test]
        public void testSayHelloTo()
        {
            Assert.AreEqual("Hello Computer", sampleLibClass.sayHelloTo("Computer"), "Regular say hello
        }

        [Test]
        public void testSayHelloToWrong()
        {
            Assert.AreEqual("Hello All", sampleLibClass.sayHelloTo("Computer"), "Regular say hello won'
        }


        [Test]
        public void testSumCorrect()
        {
            Assert.AreEqual(9, sampleLibClass.sum(4, 5), "Regular sum should work");
        }

        [Test]
```

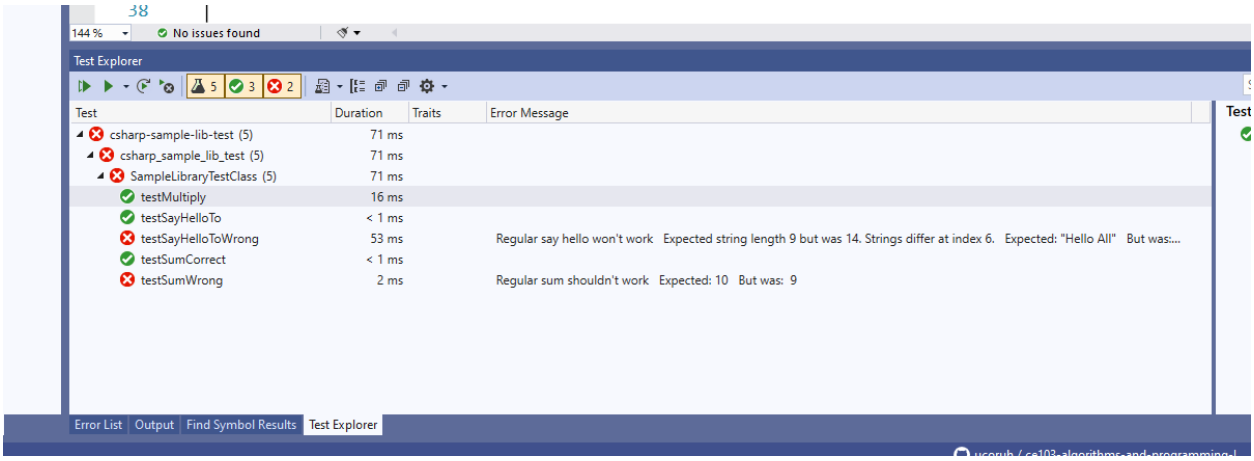
```
        public void testSumWrong()
        {
            Assert.AreEqual(10, sampleLibClass.sum(4, 5), "Regular sum shouldn't work");
        }

        [Test]
        public void testMultiply()
        {
            Assert.AreEqual(20, sampleLib.multiply(4, 5), "Regular multiplication should work");
        }
    }
}
```

---

sample class library

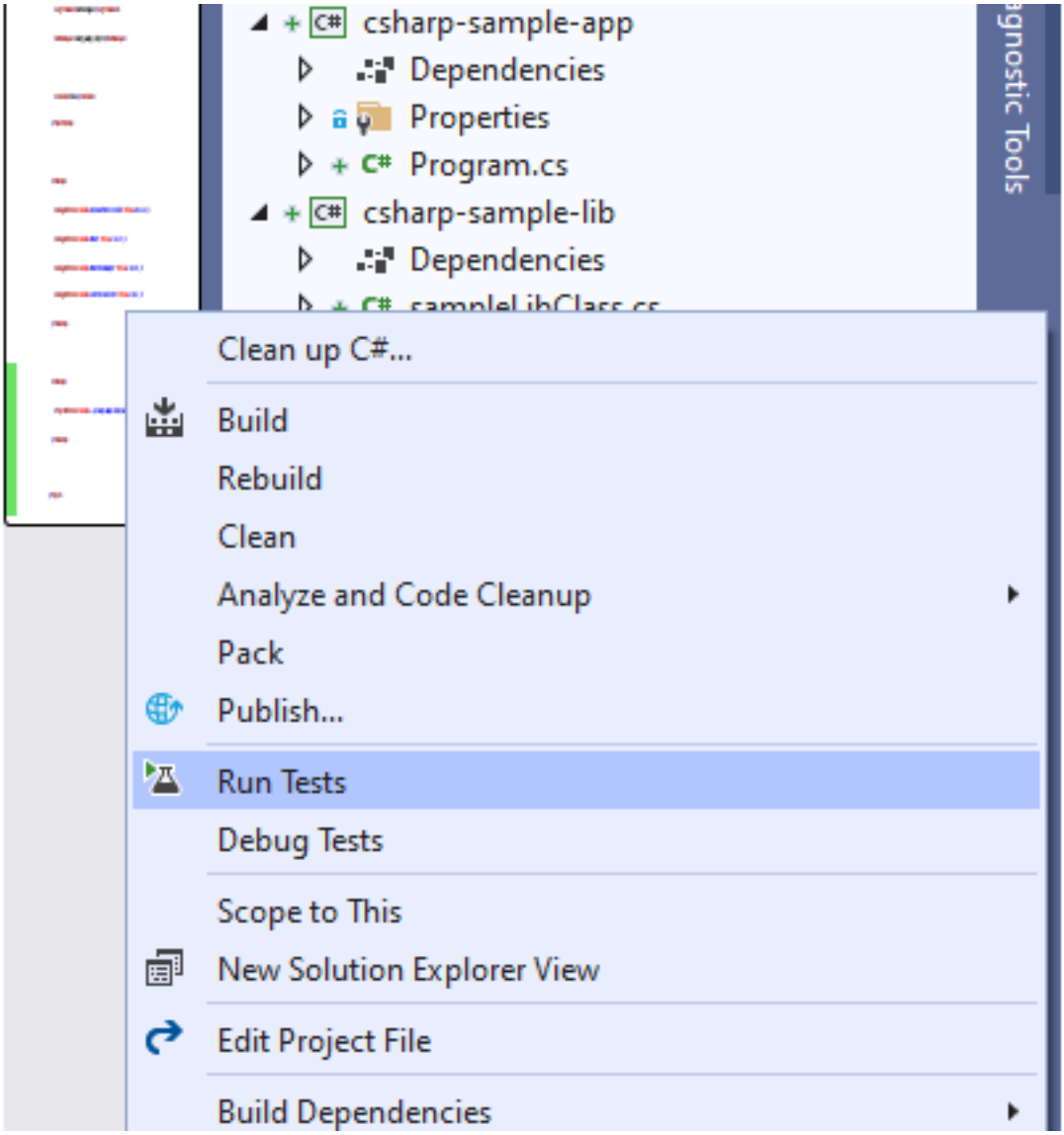
```
using System;

namespace csharp_sample_lib
{
    public class sampleLibClass
    {
        public static string sayHelloTo(string name)
        {
            string result = String.Empty;

            if (!String.IsNullOrEmpty(name))
            {
                result = "Hello " + name;
            }
            else
            {
                result = "Hello There";
            }

            Console.WriteLine(result);

            return result;
        }

        public static int sum(int a, int b)
        {
            int c = 0;
            c = a + b;
            return c;
        }

        public int multiply(int a, int b)
        {
            return a * b;
        }
    }
}
```

Open test explorer and run tests



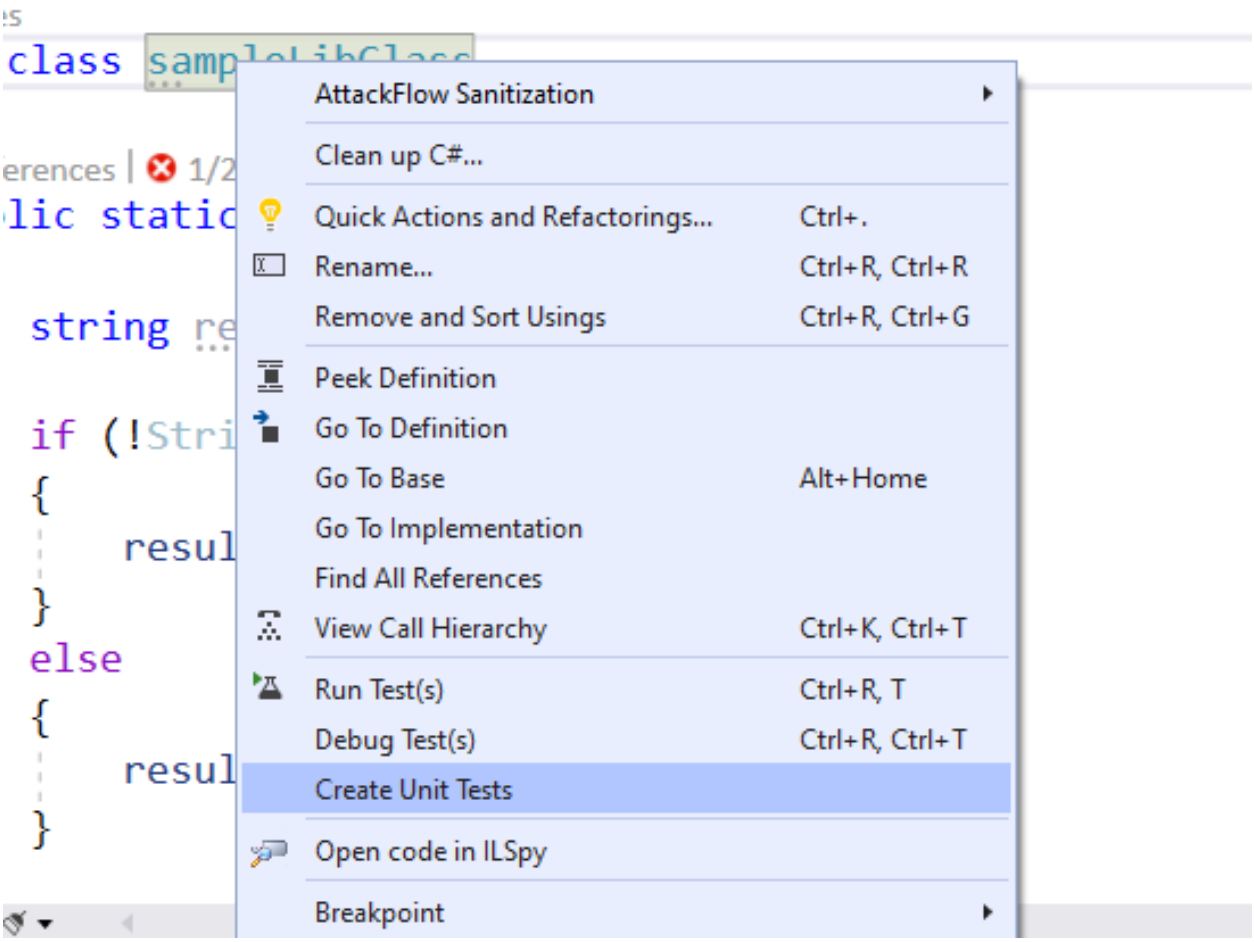
---

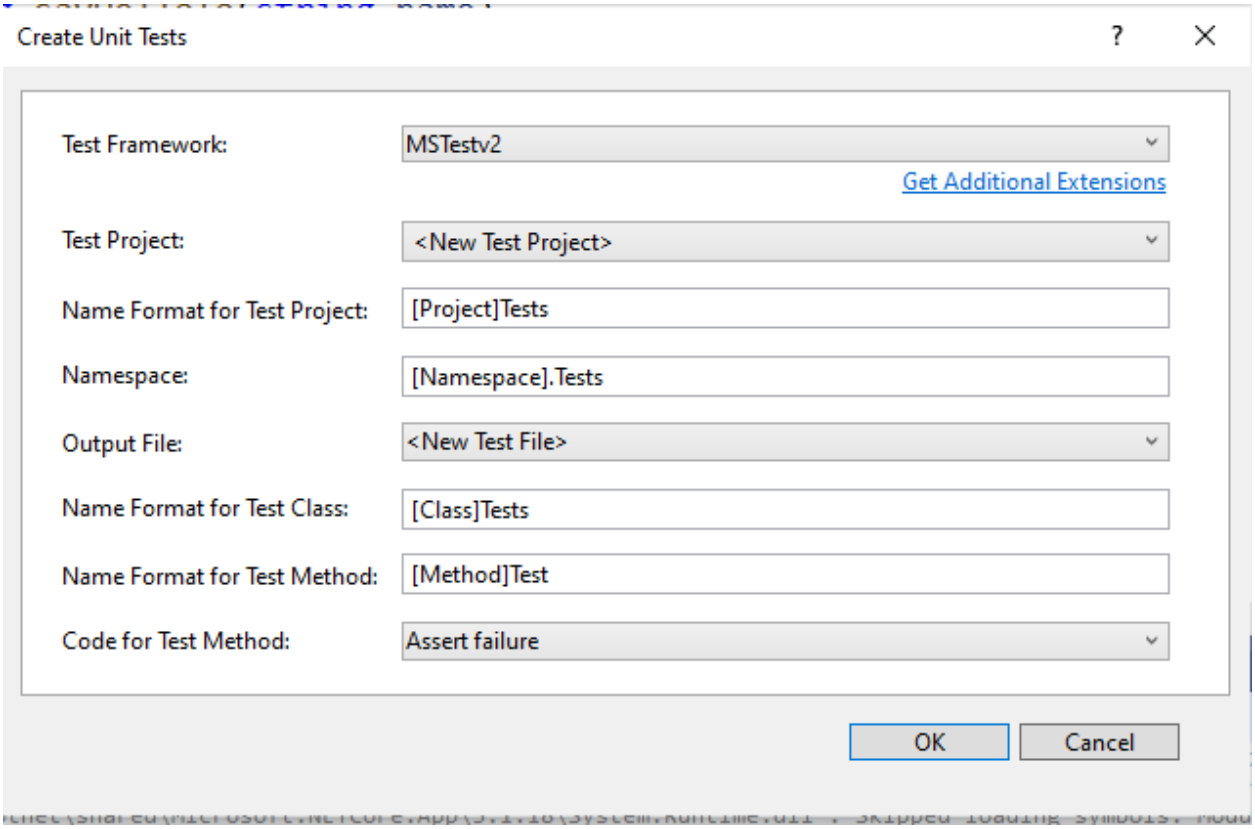
or you can run from project

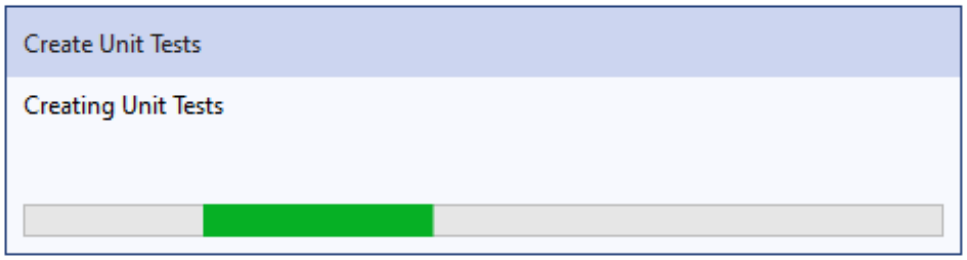


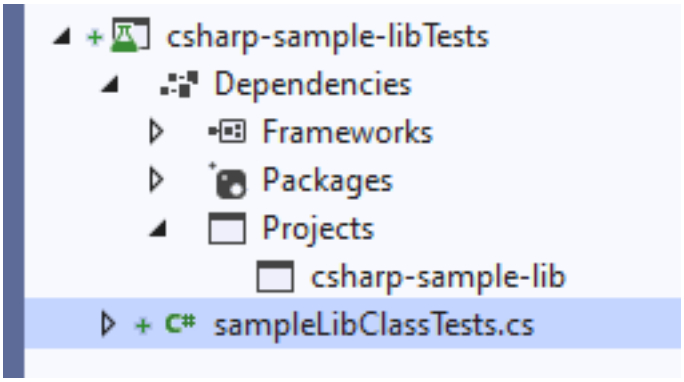
Also we can create unit test from library class,

right click the sampleLibClass and select create unit tests but this option do not provide nunit tests.




Create Unit Tests
Test Framework:
MSTestv2
Get Additional Extensions
Test Project:
<New Test Project>
Name Format for Test Project:
[Project]Tests
Namespace:
[Namespace].Tests
Output File:
<New Test File>
Name Format for Test Class:
[Class]Tests
Name Format for Test Method:
[Method]Test
Code for Test Method:
Assert failure
OK
Cancel


Create Unit Tests
Creating Unit Tests

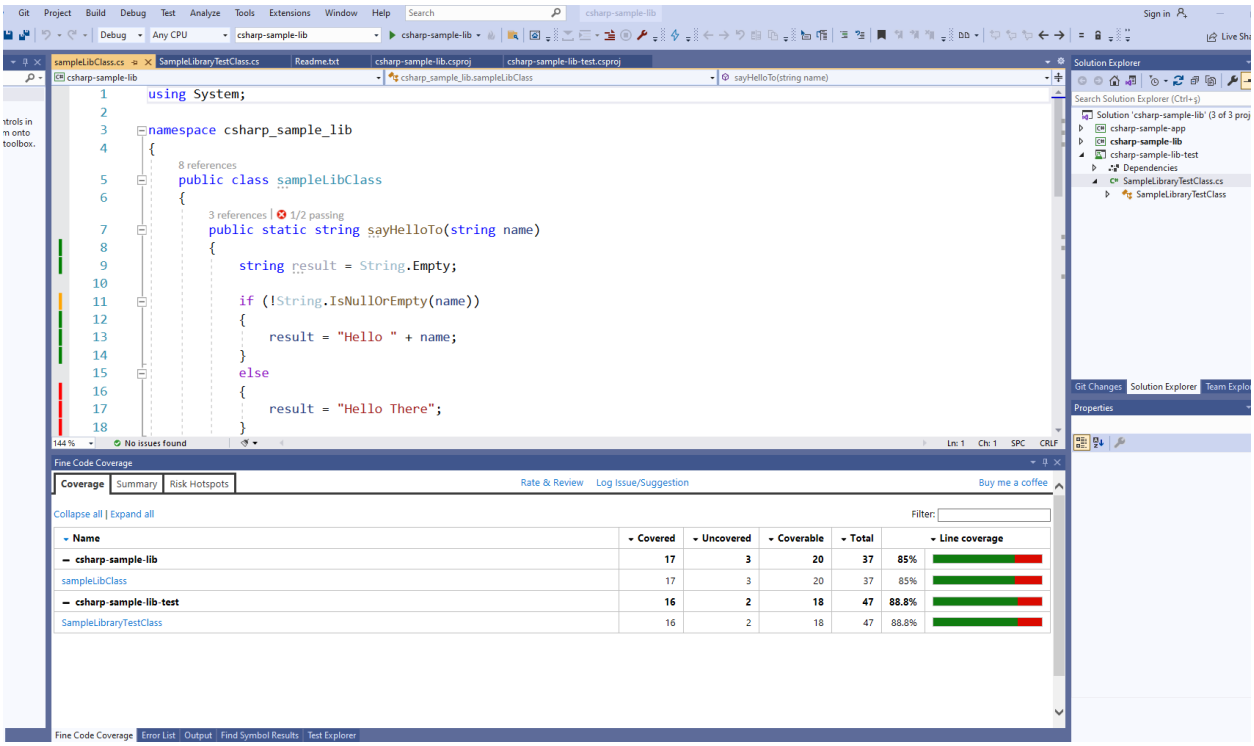

csharp-sample-libTests
Dependencies
Frameworks
Packages
Projects
csharp-sample-lib
sampleLibClassTests.cs

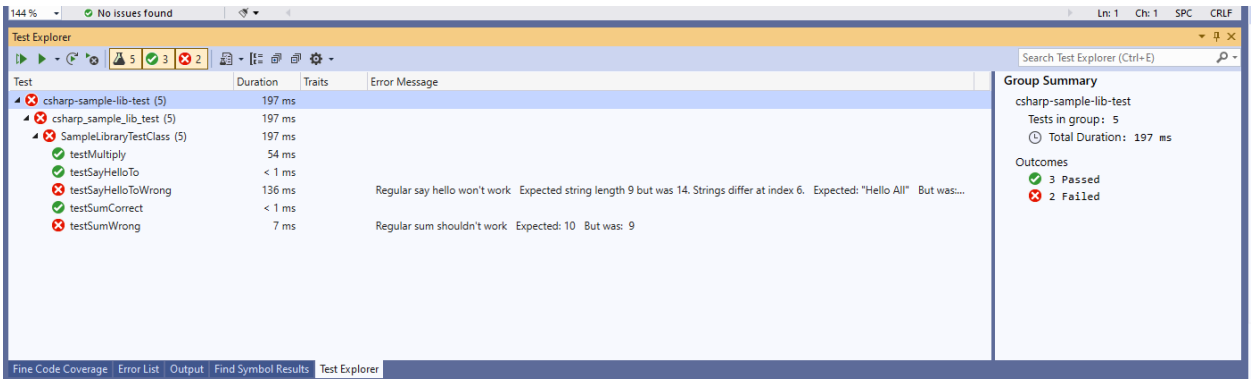
```
using Microsoft.VisualStudio.TestTools.UnitTesting;
using csharp_sample_lib;
using System;
using System.Collections.Generic;
using System.Text;

namespace csharp_sample_lib.Tests
{
    [TestClass()]
    public class sampleLibClassTests
    {
        [TestMethod()]
        public void sayHelloToTest()
        {
            Assert.Fail();
        }

        [TestMethod()]
        public void sumTest()
        {
            Assert.Fail();
        }

        [TestMethod()]
        public void multiplyTest()
        {
            Assert.Fail();
        }
    }
}
```

---

we will not commit this changes and continue from nunit test project, the fine code coverage also work for nunit test but not provide inline highlighting

if we run tests we will have the following outputs





Inline code highlight is part of enterprise visual studio edition

Analyzing code coverage in Visual Studio - DEV Community[13]

1 TL;DR

Additional information you can use OpenCover + Nunit Runner + Report Generator together to setup a code coverage report but it has complex batch running process. After a few try I decided to use fine code coverage but here is the usage not tested well.

First unit test runner tool doesn't support .Net Core

c# - The NUnit 3 driver encountered an error while executing reflected code (NUnit.Engine.NUnitEngineException) - Stack Overflow[14]

Follow the instructions on the link

[13] https://dev.to/rruizdev/analizando-cobertura-del-codigo-en-visual-studio-1p27

[14] https://stackoverflow.com/questions/64611083/the-nunit-3-driver-encountered-an-error-while-executing-reflected-code-nunit-en

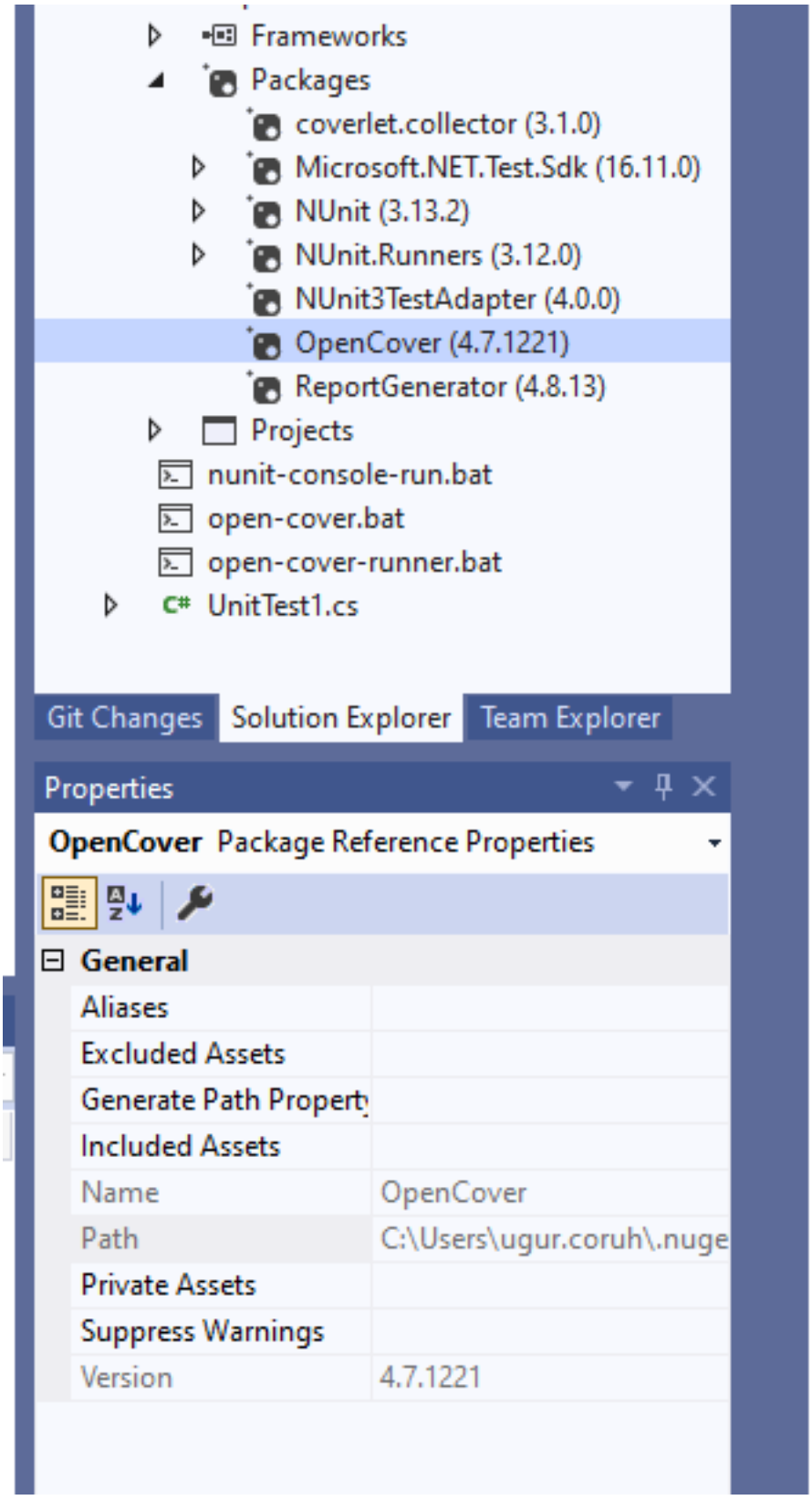
CMD OpenCover · sukhoi1/Useful-Notes Wiki · GitHub[15]

Install OpenCover, ReportGenerator, Nunit,Runners packages then use the package installation folder to get tools that you need

---

Here is a sample for open cover, select package and copy path

[15] https://github.com/sukhoi1/Useful-Notes/wiki/CMD-OpenCover



Goto path and tools

`C:\Users\ugur.coruh\.nuget\packages\opencover\4.7.1221`

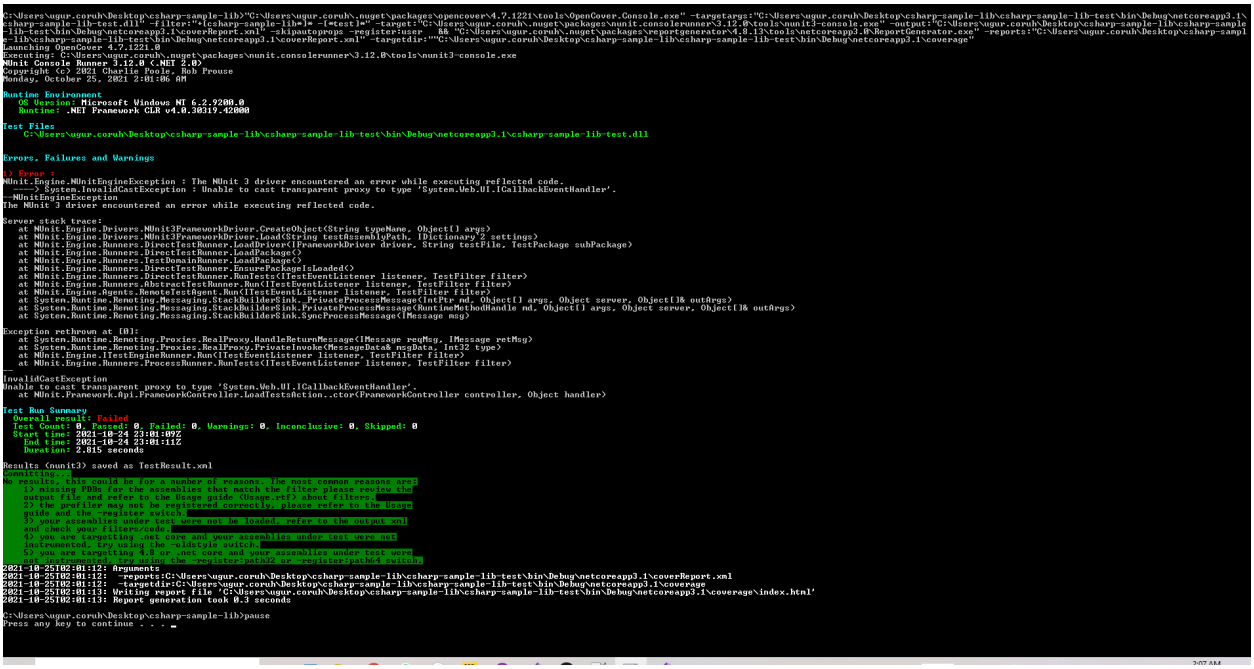
You need to setup some batch similar with following

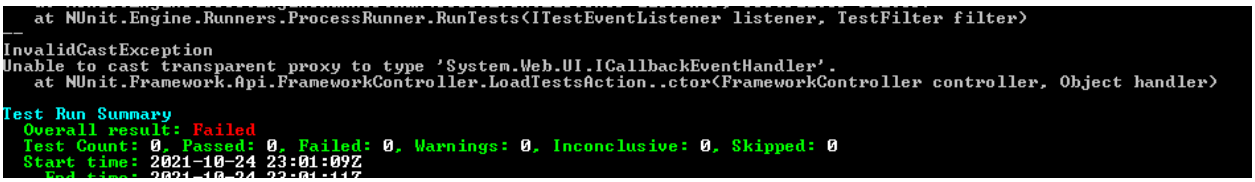
run-test-coverage.bat

```
set pathA=C:\Users\ugur.coruh\.nuget\packages\opencover\4.7.1221\tools
set pathB=C:\Users\ugur.coruh\.nuget\packages\nunit.consolerunner\3.12.0\tools
set pathC=C:\Users\ugur.coruh\.nuget\packages\reportgenerator\4.8.13\tools\netcoreapp3.0
set dllpath=C:\Users\ugur.coruh\Desktop\csharp-sample-lib\csharp-sample-lib-test\bin\Debug\netcoreapp3.

"%pathA%\OpenCover.Console.exe" ^
-targetargs:"%dllpath%\csharp-sample-lib-test.dll" ^
-filter:"+[csharp-sample-lib*]* -[*test]*" ^
-target:"%pathB%\nunit3-console.exe" ^
-output:"%dllpath%\coverReport.xml" ^
-skipautoprops -register:user && "%pathC%\ReportGenerator.exe" -reports:"%dllpath%\coverReport.xml" -ta
pause
```

---

but nunit3-console.exe gives error





---

for this compatibility issues I prefer to use fine code coverage extension.

OpenCover related studies

Code coverage of manual or automated tests with OpenCover for .NET applications – Automation Rhapsody[16]

Code coverage of .NET Core unit tests with OpenCover – Automation Rhapsody[17]

[16] https://automationrhapsody.com/code-coverage-manual-automated-tests-opencover-net-applications/
[17] https://automationrhapsody.com/code-coverage-net-core-unit-tests-opencover/

Sample OpenCover report

Summary - Coverage Report[18]

1.0.1 Download and Setup OpenCover, NUnit Console, Report Generator without Package Manager

You can also download the tools from github project pages and install on your operating system,

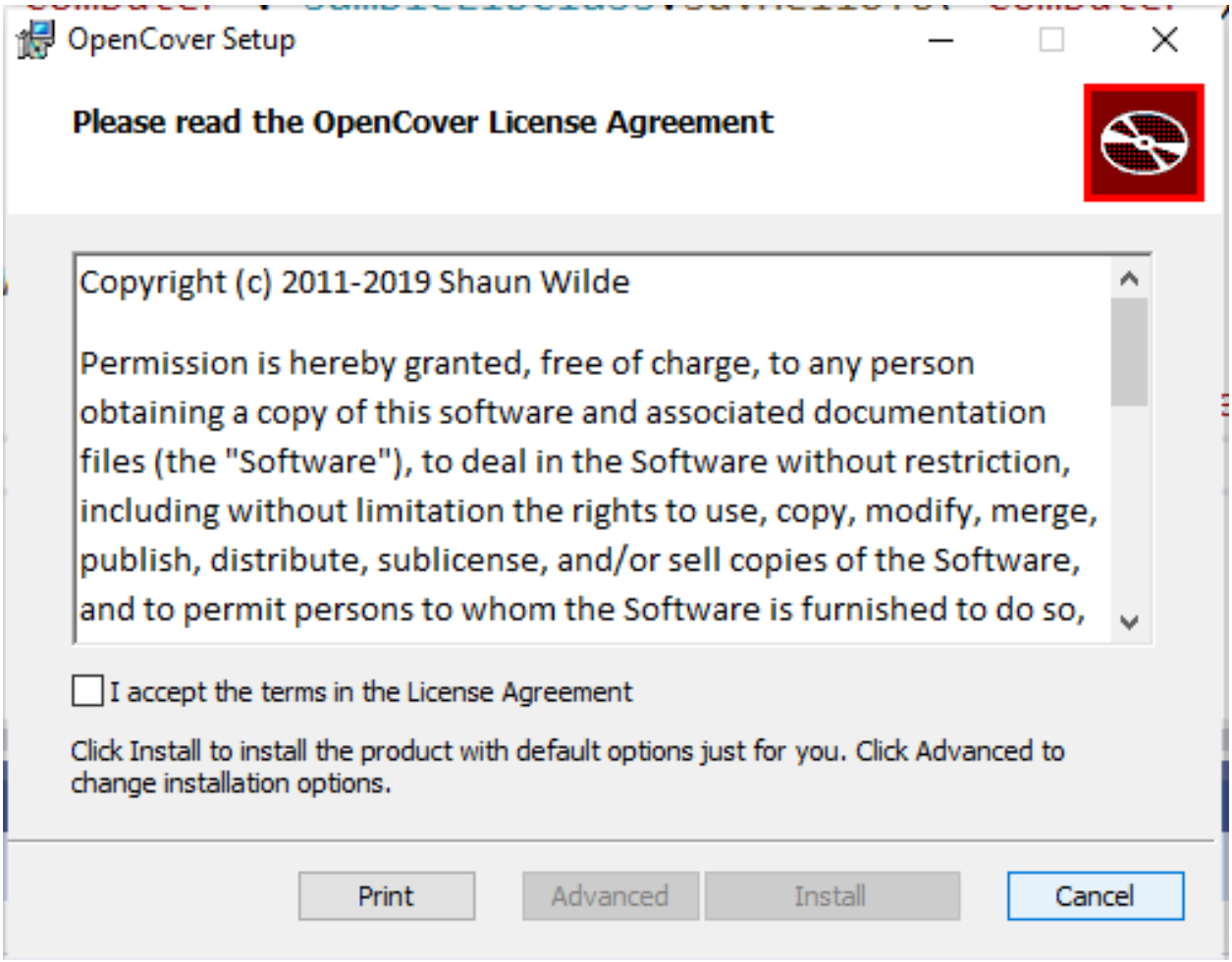
---

1.0.2 OpenCover

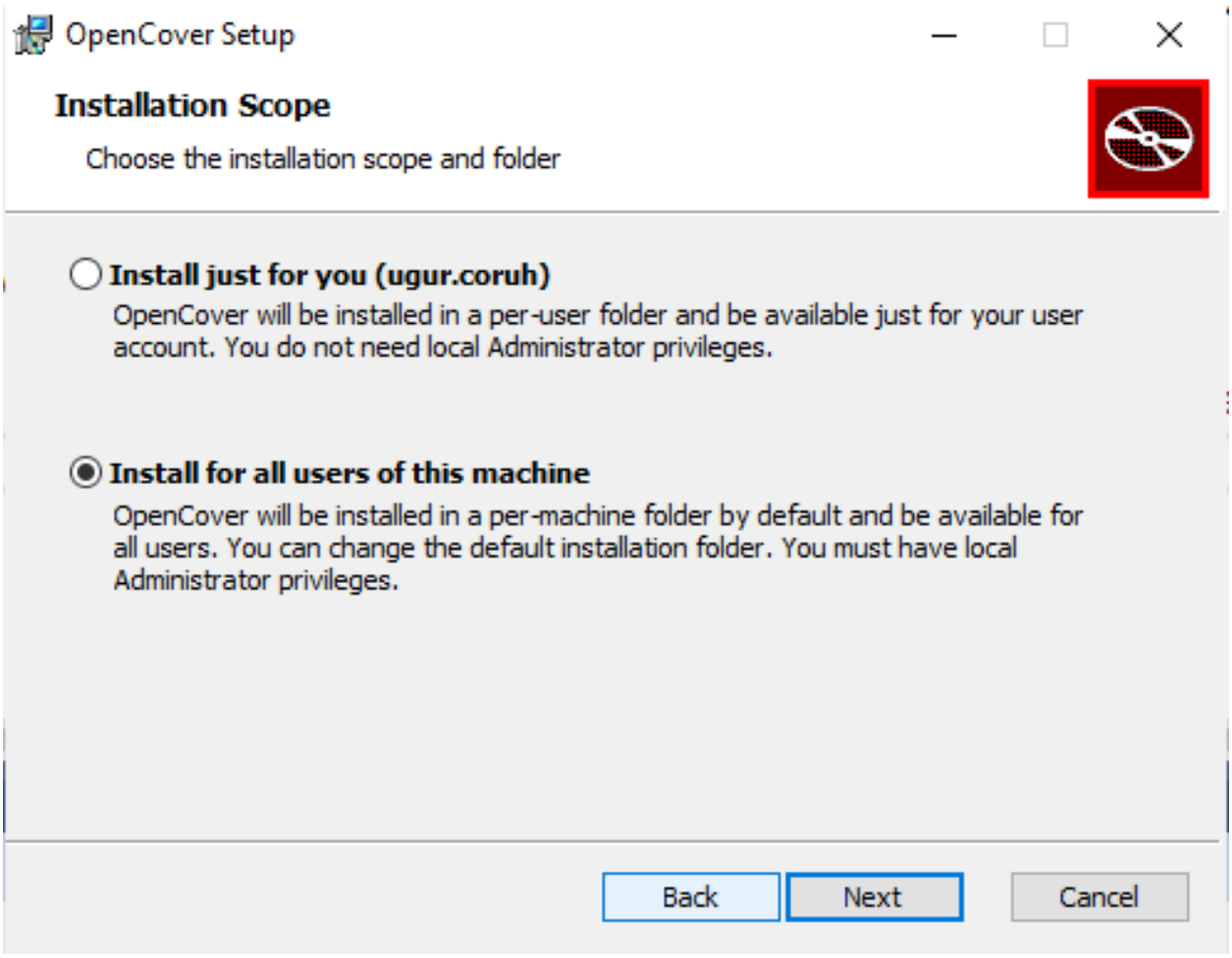
Releases · OpenCover/opencover · GitHub[19]

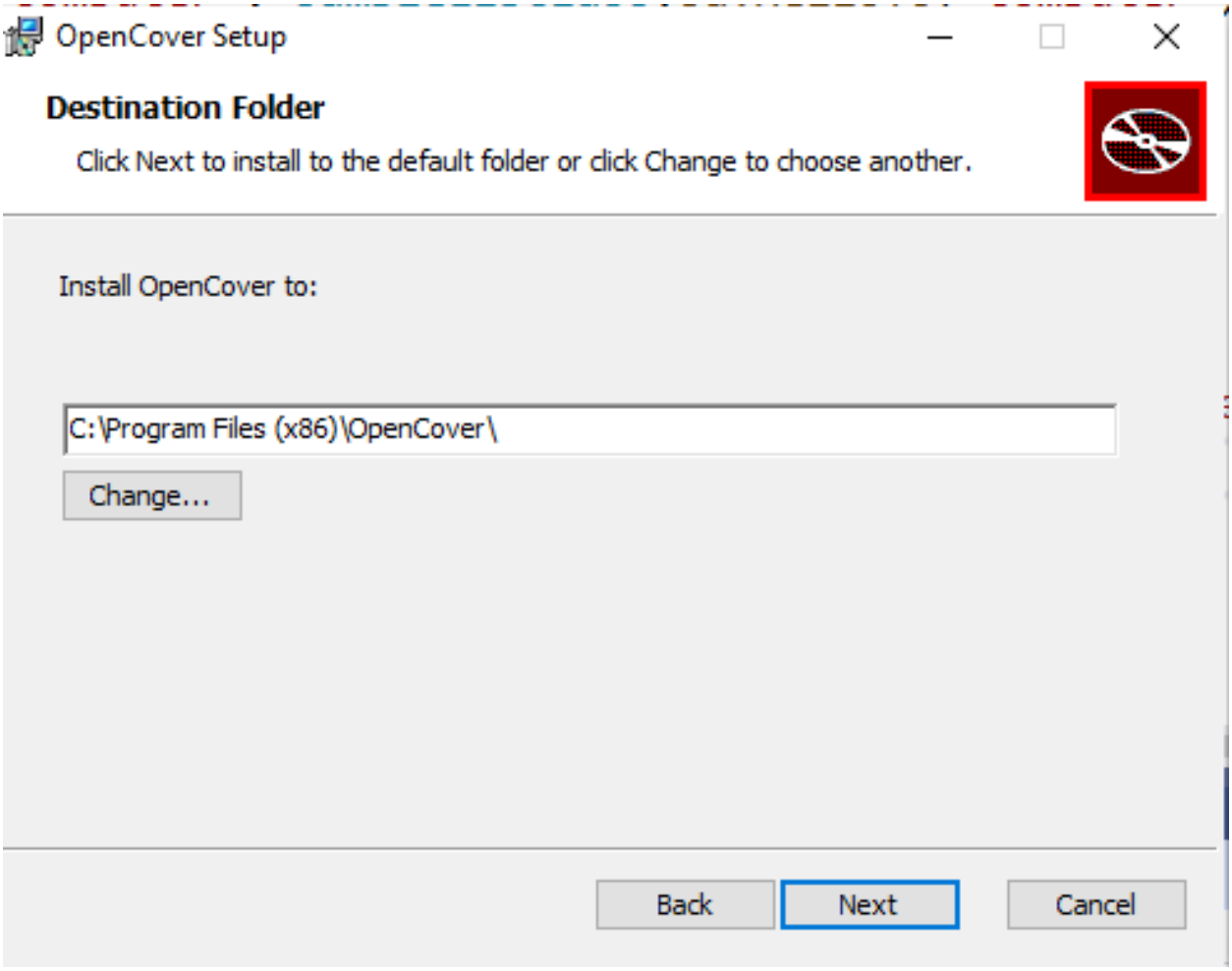
---

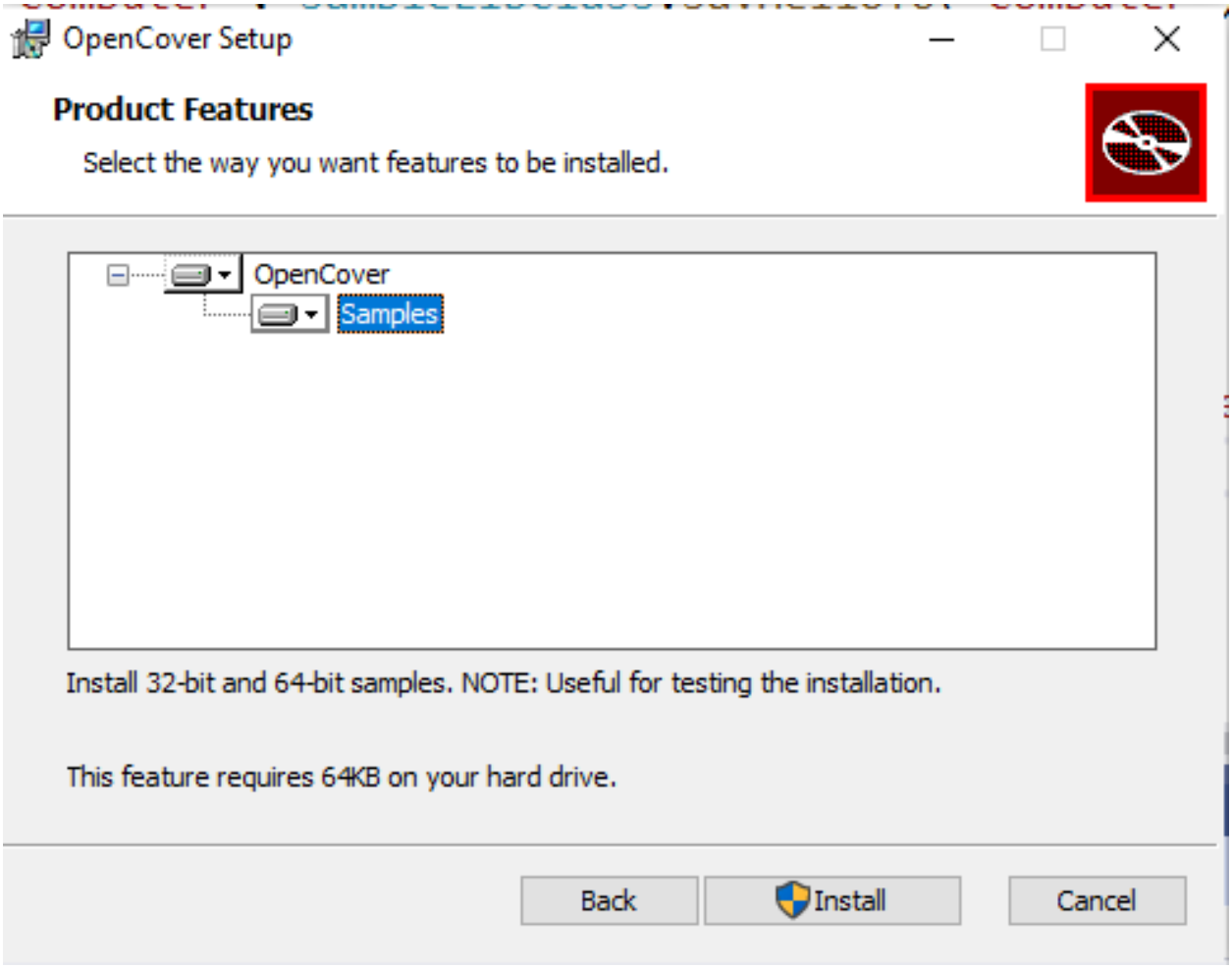
[18] https://automationrhapsody.com/examples/OpenCover-report/
[19] https://github.com/OpenCover/opencover/releases

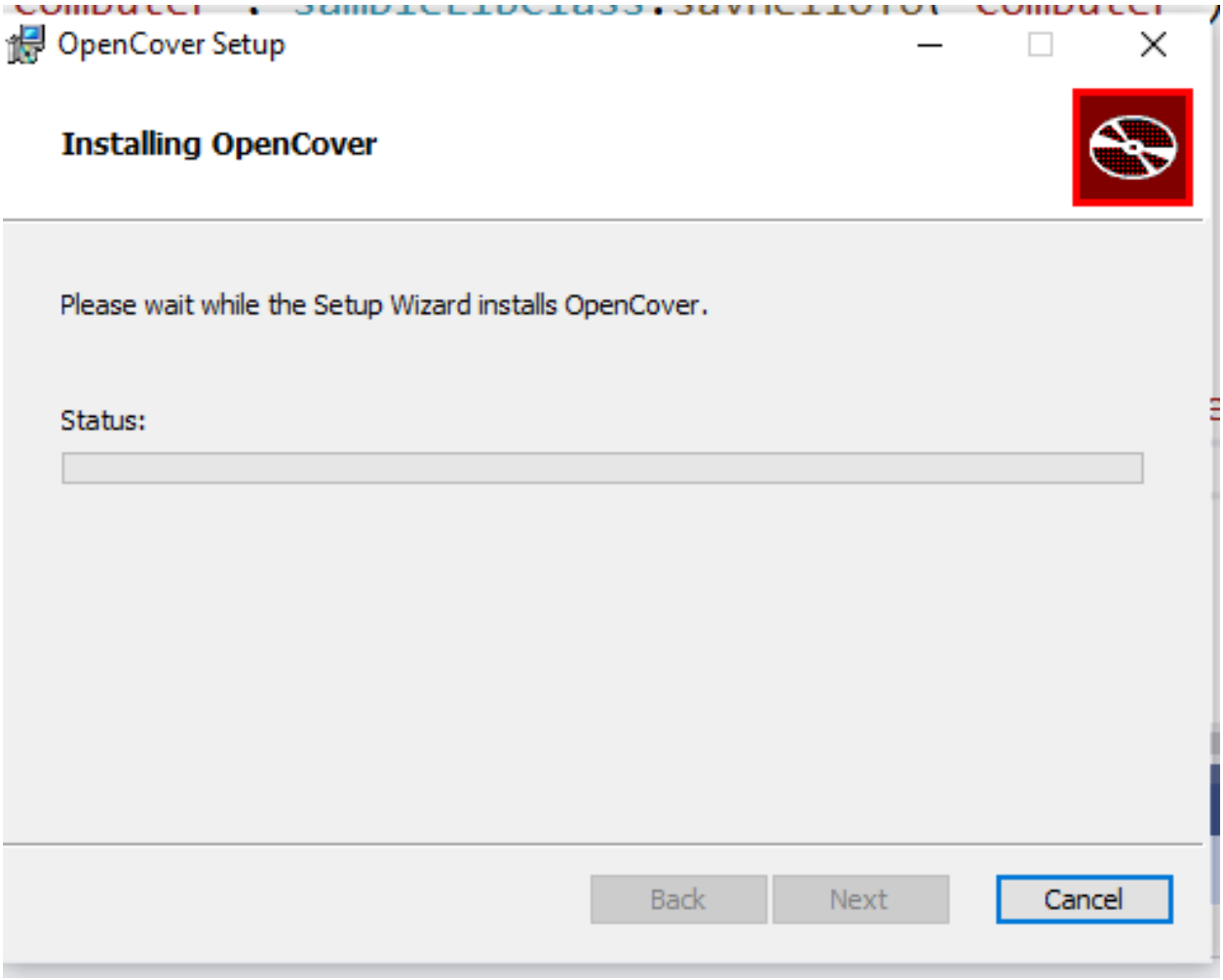


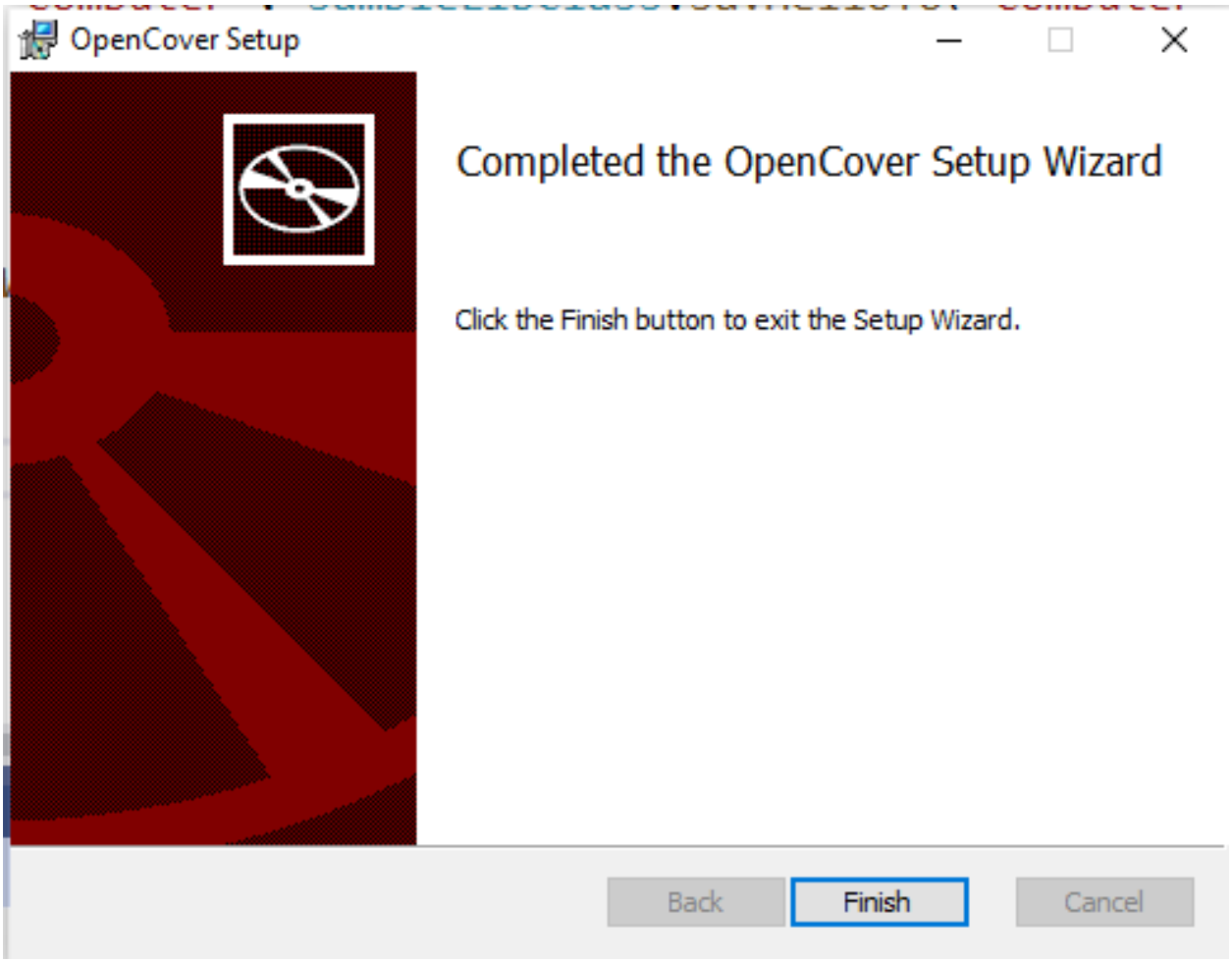
Select advanced and then install for all users

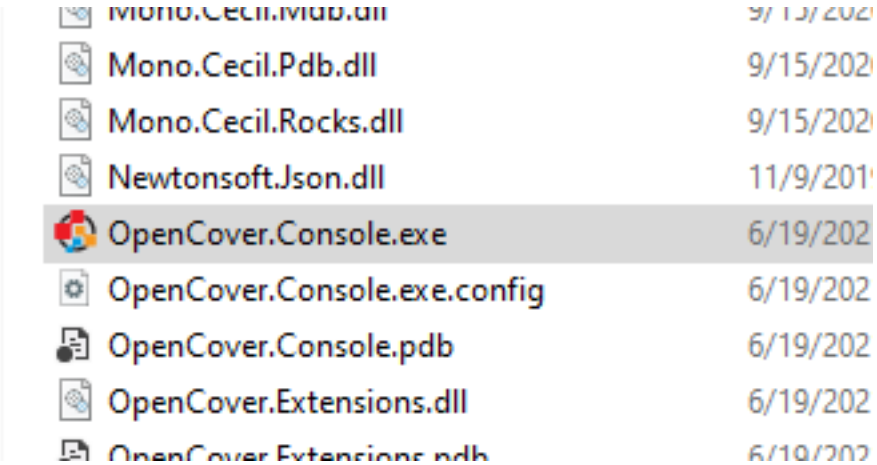

OpenCover Setup
Installation Scope
Choose the installation scope and folder
Install just for you (ugur.coruh)
OpenCover will be installed in a per-user folder and be available just for your user account. You do not need local Administrator privileges.
Install for all users of this machine
OpenCover will be installed in a per-machine folder by default and be available for all users. You can change the default installation folder. You must have local Administrator privileges.
Back
Next
Cancel

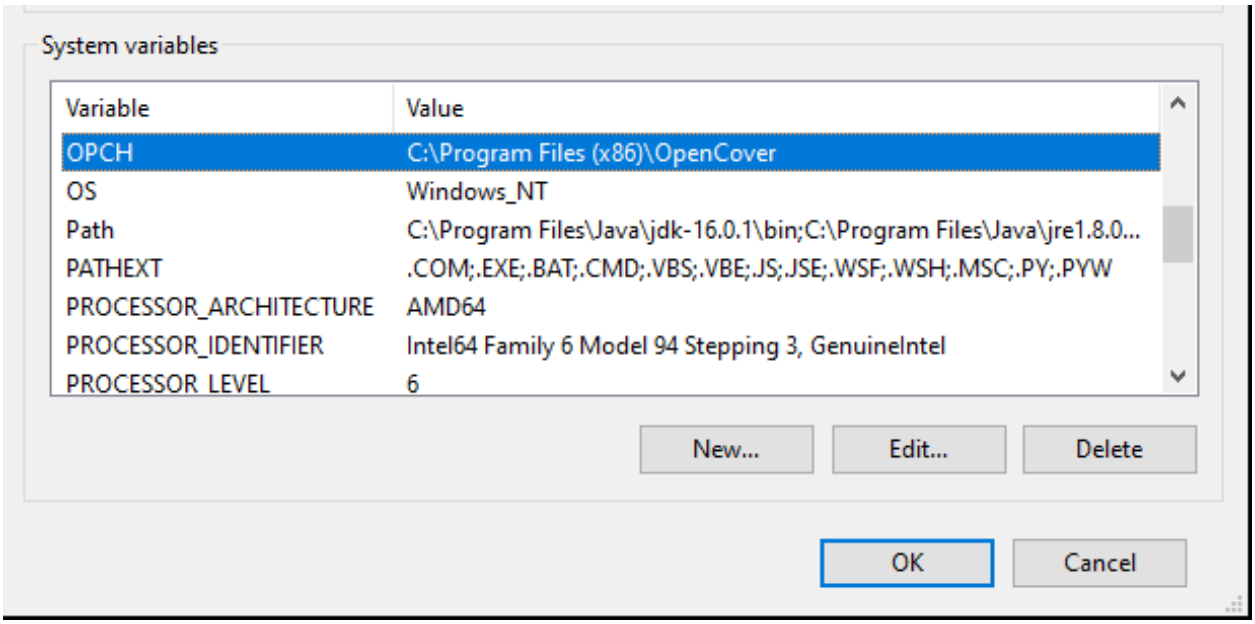

OpenCover Setup
Destination Folder
Click Next to install to the default folder or click Change to choose another.
Install OpenCover to:
C:\Program Files (x86)\OpenCover\
Change...
Back
Next
Cancel

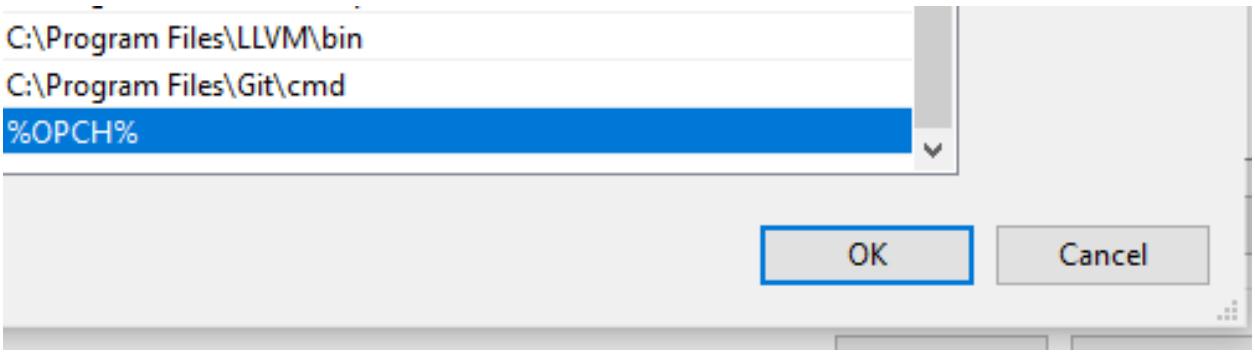

OpenCover Setup
Product Features
Select the way you want features to be installed.
OpenCover
Samples
Install 32-bit and 64-bit samples. NOTE: Useful for testing the installation.
This feature requires 64KB on your hard drive.
Back
Install
Cancel

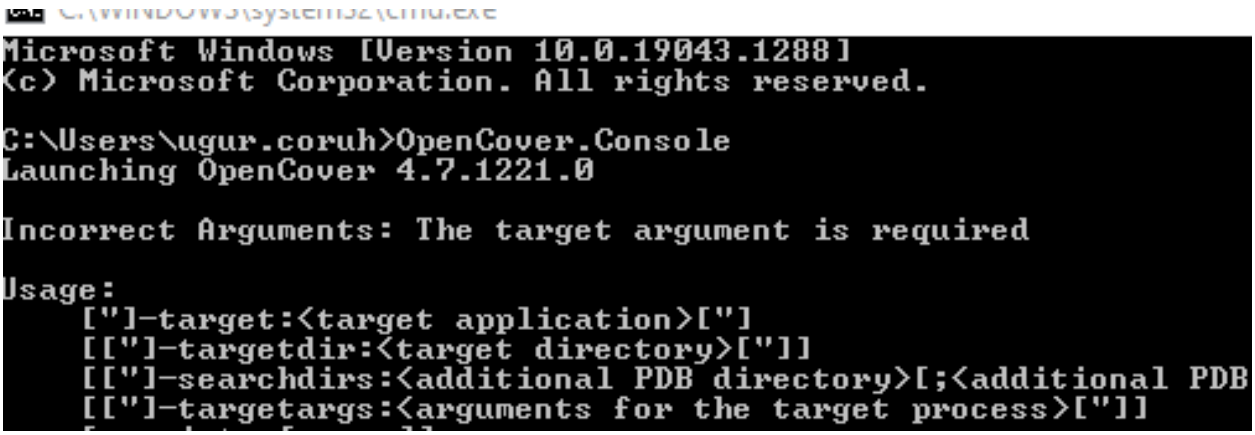

OpenCover Setup
Installing OpenCover
Please wait while the Setup Wizard installs OpenCover.
Status:
Back
Next
Cancel


OpenCover Setup
Completed the OpenCover Setup Wizard
Click the Finish button to exit the Setup Wizard.
Back
Finish
Cancel


Mono.Cecil.Pdb.dll
Mono.Cecil.Rocks.dll
Newtonsoft.Json.dll
OpenCover.Console.exe
OpenCover.Console.exe.config
OpenCover.Console.pdb
OpenCover.Extensions.dll



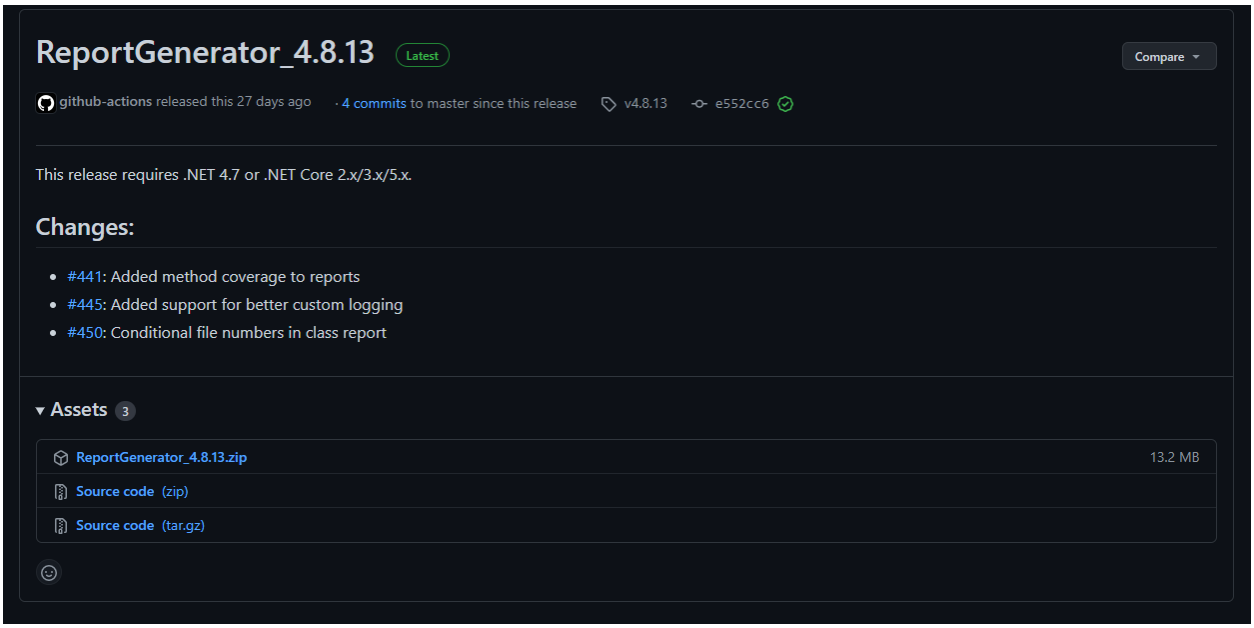


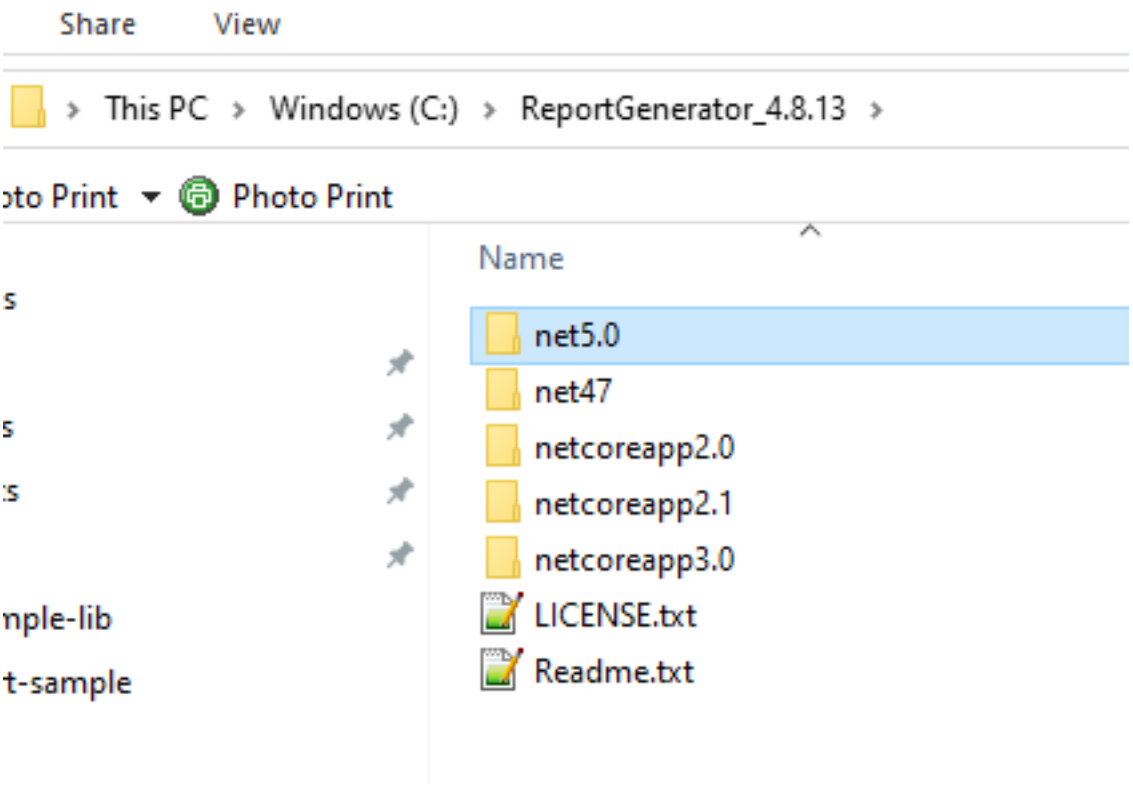


1.0.3 ReportGenerator

Release ReportGenerator_4.8.13 · danielpalme/ReportGenerator · GitHub[20]

[20] https://github.com/danielpalme/ReportGenerator/releases/tag/v4.8.13

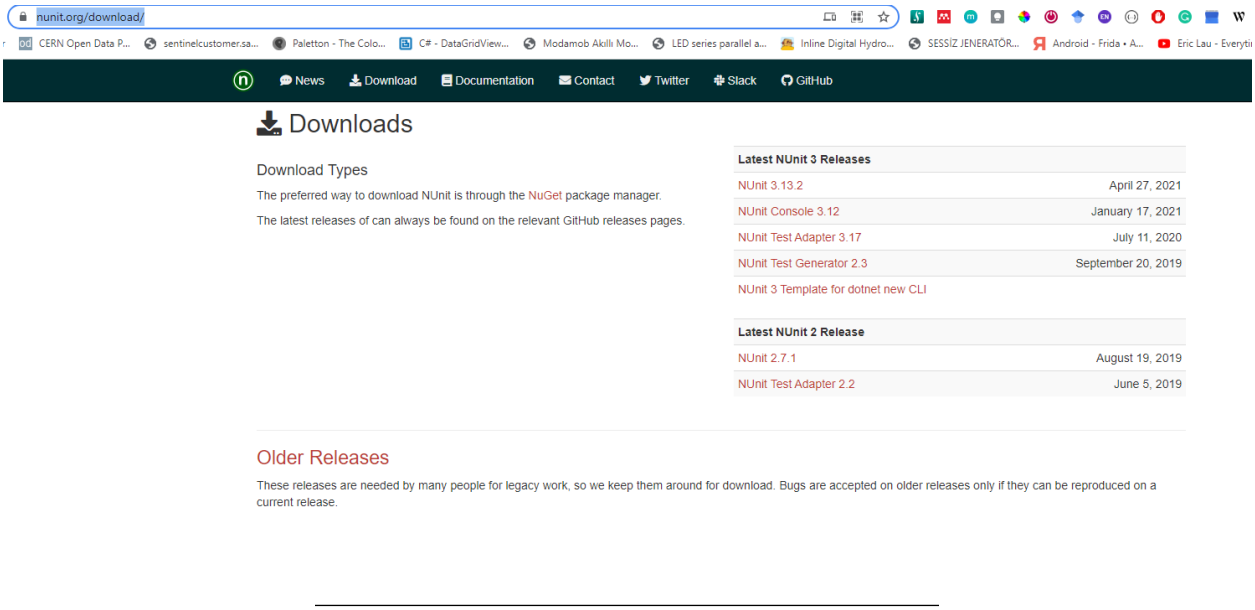


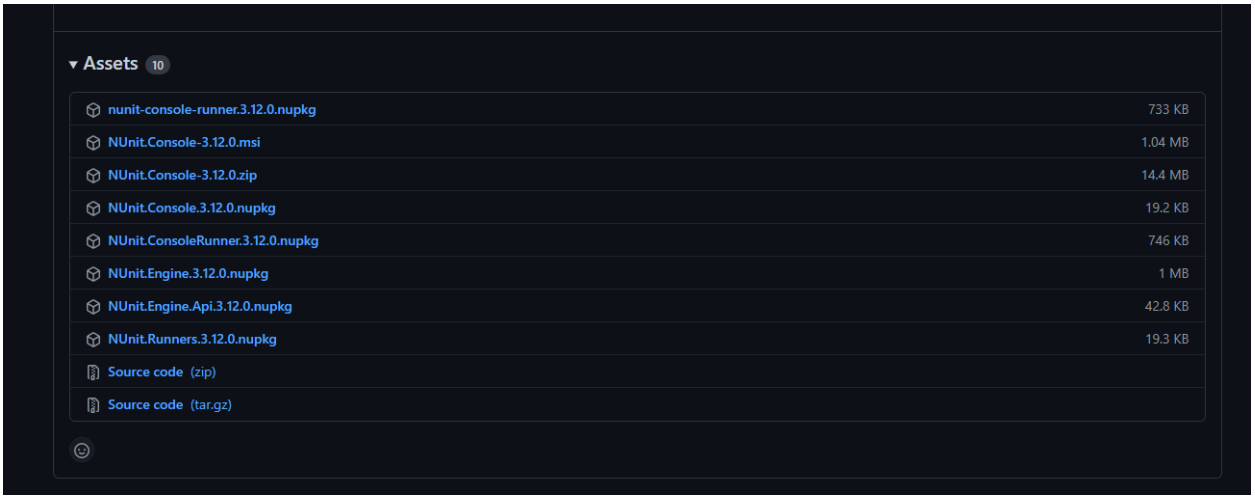


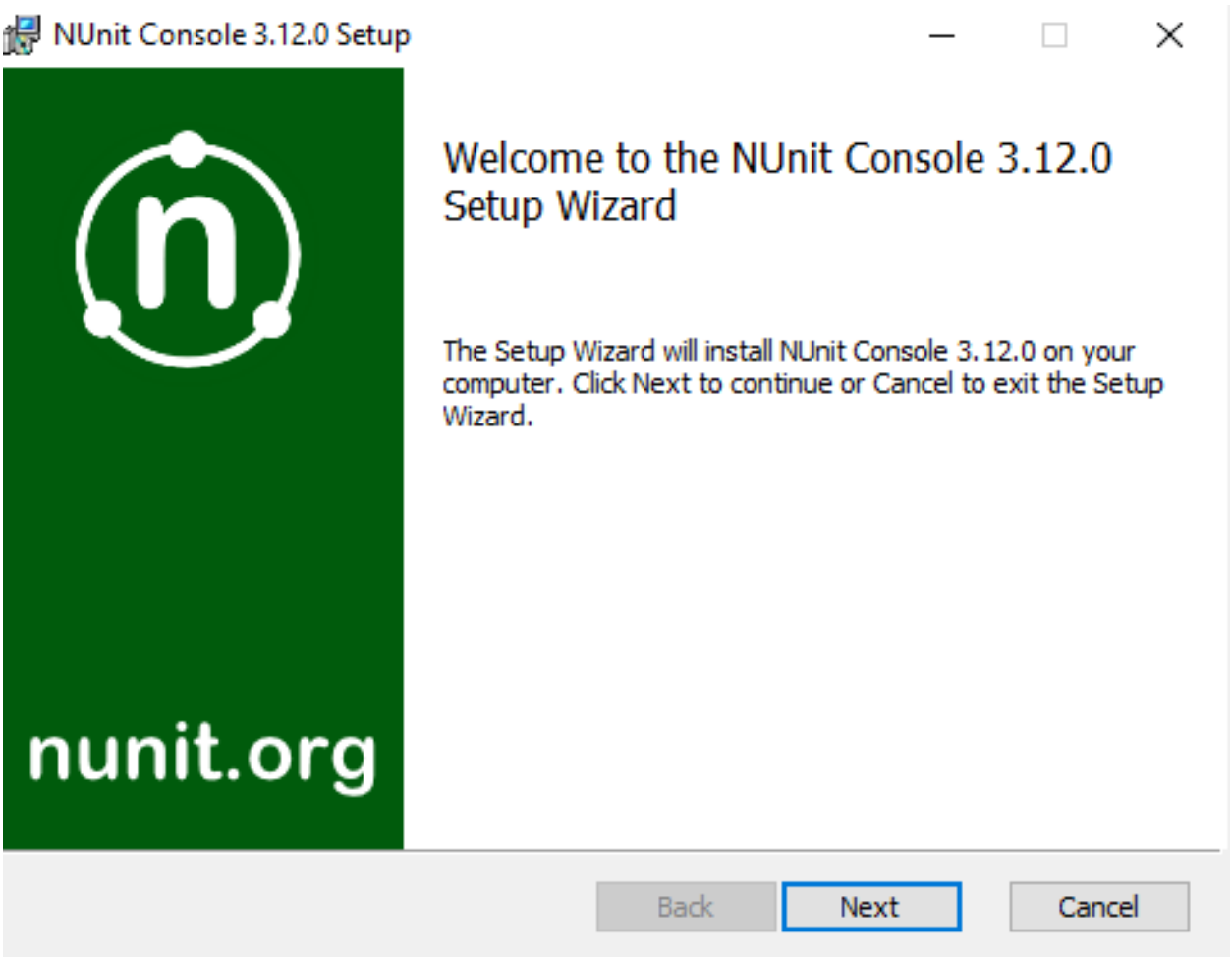
1.0.4 NUnit Console

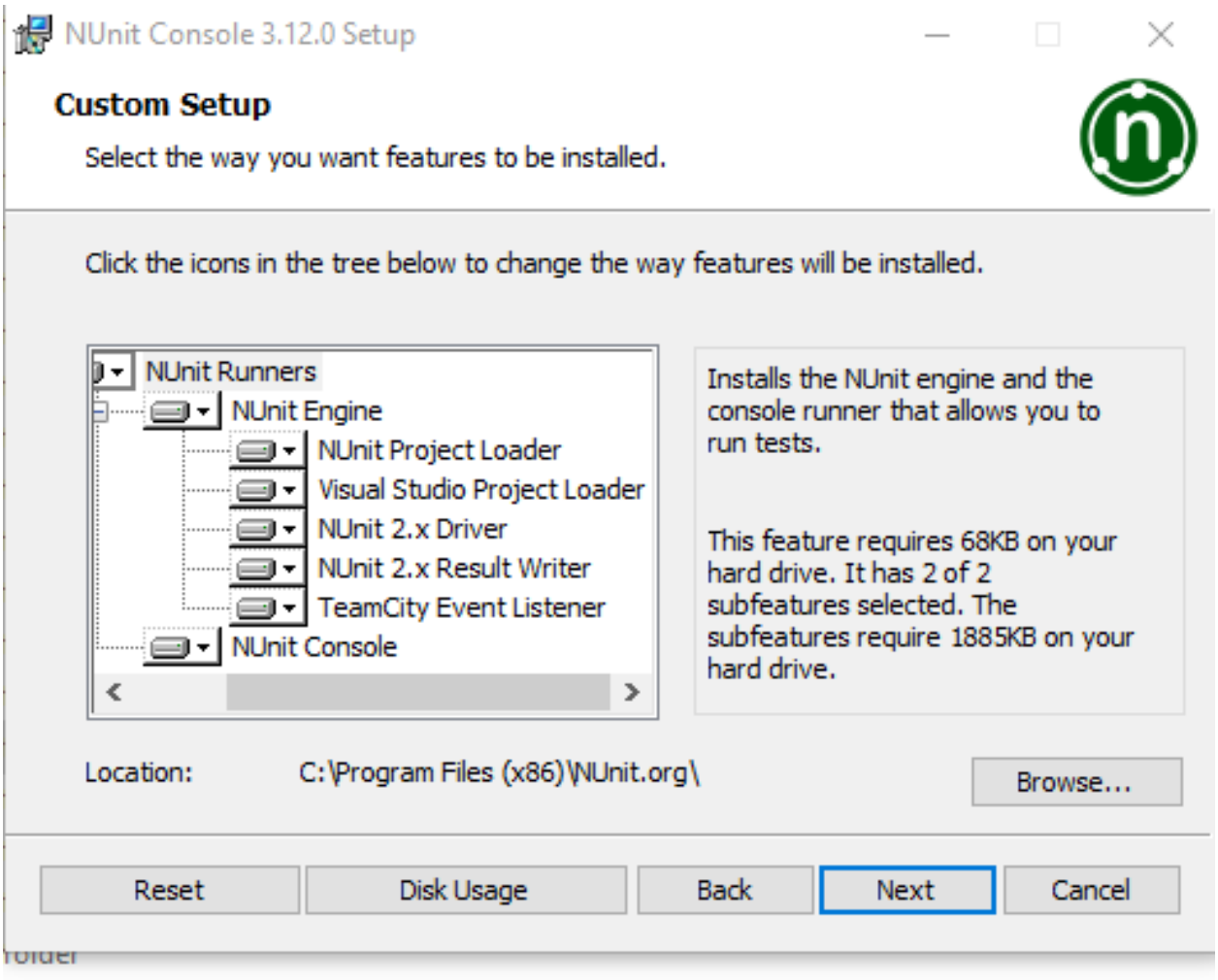
Downloads[21]

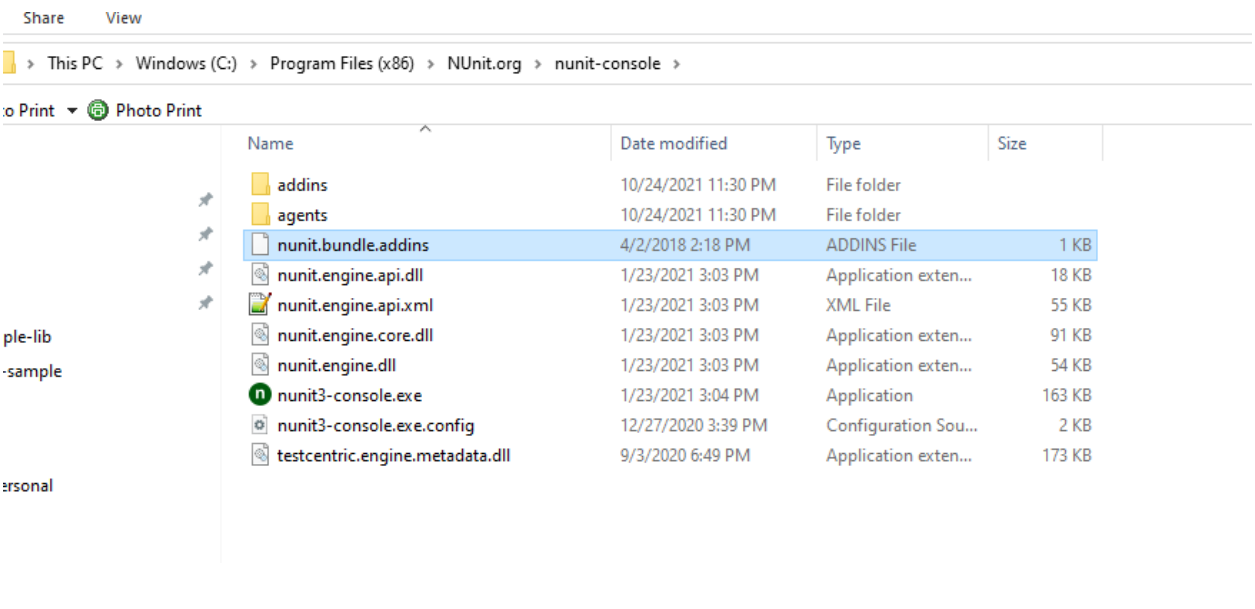
[21] https://nunit.org/download/


nunit.org/download/
News
Download
Documentation
Contact
Twitter
Slack
GitHub
Downloads
Download Types
The preferred way to download NUnit is through the NuGet package manager.
The latest releases of can always be found on the relevant GitHub releases pages.
Latest NUnit 3 Releases
NUnit 3.13.2 April 27, 2021
NUnit Console 3.12 January 17, 2021
NUnit Test Adapter 3.17 July 11, 2020
NUnit Test Generator 2.3 September 20, 2019
NUnit 3 Template for dotnet new CLI
Latest NUnit 2 Release
NUnit 2.7.1 August 19, 2019
NUnit Test Adapter 2.2 June 5, 2019
Older Releases
These releases are needed by many people for legacy work, so we keep them around for download. Bugs are accepted on older releases only if they can be reproduced on a current release.


Assets 10
nunit-console-runner.3.12.0.nupkg 733 KB
NUnit.Console-3.12.0.msi 1.04 MB
NUnit.Console-3.12.0.zip 14.4 MB
NUnit.Console.3.12.0.nupkg 19.2 KB
NUnit.ConsoleRunner.3.12.0.nupkg 746 KB
NUnit.Engine.3.12.0.nupkg 1 MB
NUnit.Engine.Api.3.12.0.nupkg 42.8 KB
NUnit.Runners.3.12.0.nupkg 19.3 KB
Source code (zip)
Source code (tar.gz)


NUnit Console 3.12.0 Setup
Welcome to the NUnit Console 3.12.0 Setup Wizard
The Setup Wizard will install NUnit Console 3.12.0 on your computer. Click Next to continue or Cancel to exit the Setup Wizard.
nunit.org
Back
Next
Cancel


NUnit Console 3.12.0 Setup
Custom Setup
Select the way you want features to be installed.
Click the icons in the tree below to change the way features will be installed.
NUnit Runners
NUnit Engine
NUnit Project Loader
Visual Studio Project Loader
NUnit 2.x Driver
NUnit 2.x Result Writer
TeamCity Event Listener
NUnit Console
Installs the NUnit engine and the console runner that allows you to run tests.
This feature requires 68KB on your hard drive. It has 2 of 2 subfeatures selected. The subfeatures require 1885KB on your hard drive.
Location:
C:\Program Files (x86)\NUnit.org\
Browse...
Reset
Disk Usage
Back
Next
Cancel


Share
View
This PC
Windows (C:)
Program Files (x86)
NUnit.org
nunit-console
Photo Print
Name
Date modified
Type
Size
addins 10/24/2021 11:30 PM File folder
agents 10/24/2021 11:30 PM File folder
nunit.bundle.addins 4/2/2018 2:18 PM ADDINS File 1 KB
nunit.engine.api.dll 1/23/2021 3:03 PM Application exten... 18 KB
nunit.engine.api.xml 1/23/2021 3:03 PM XML File 55 KB
nunit.engine.core.dll 1/23/2021 3:03 PM Application exten... 91 KB
nunit.engine.dll 1/23/2021 3:03 PM Application exten... 54 KB
nunit3-console.exe 1/23/2021 3:04 PM Application 163 KB
nunit3-console.exe.config 12/27/2020 3:39 PM Configuration Sou... 2 KB
testcentric.engine.metadata.dll 9/3/2020 6:49 PM Application exten... 173 KB

1.0.5 NUnit + MSTest Batch Report Generation (Not Tested)

OpenCover and ReportGenerator Unit Test Coverage in Visual Studio 2013 and 2015 – CodeHelper.Net[22]

OpenCover and ReportGenerator Unit Test Coverage in Visual Studio 2013 and 2015 - CodeProject[23]

1.0.6 Java Unit Tests

1.0.6.1 Eclipse IDE (JUnit4 , JUnit5) In this sample we will create two example for similar library

Please check the following links

JUnit 5 tutorial - Learn how to write unit tests[24]

JUnit 5[25]

JUnit 5 User Guide[26]

https://www.eclemma.org/

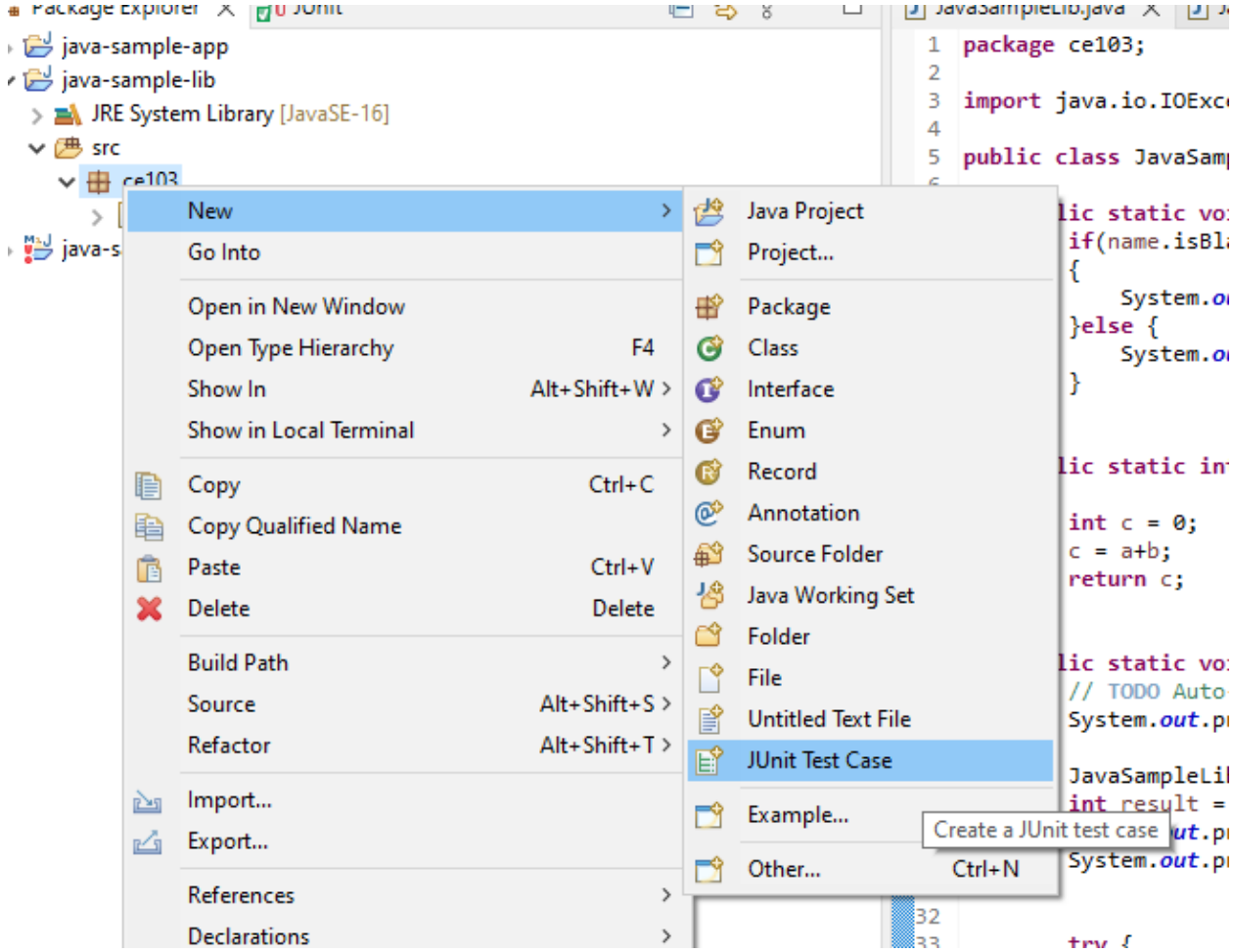
JUnit Hello World Example - Examples Java Code Geeks - 2021[27]

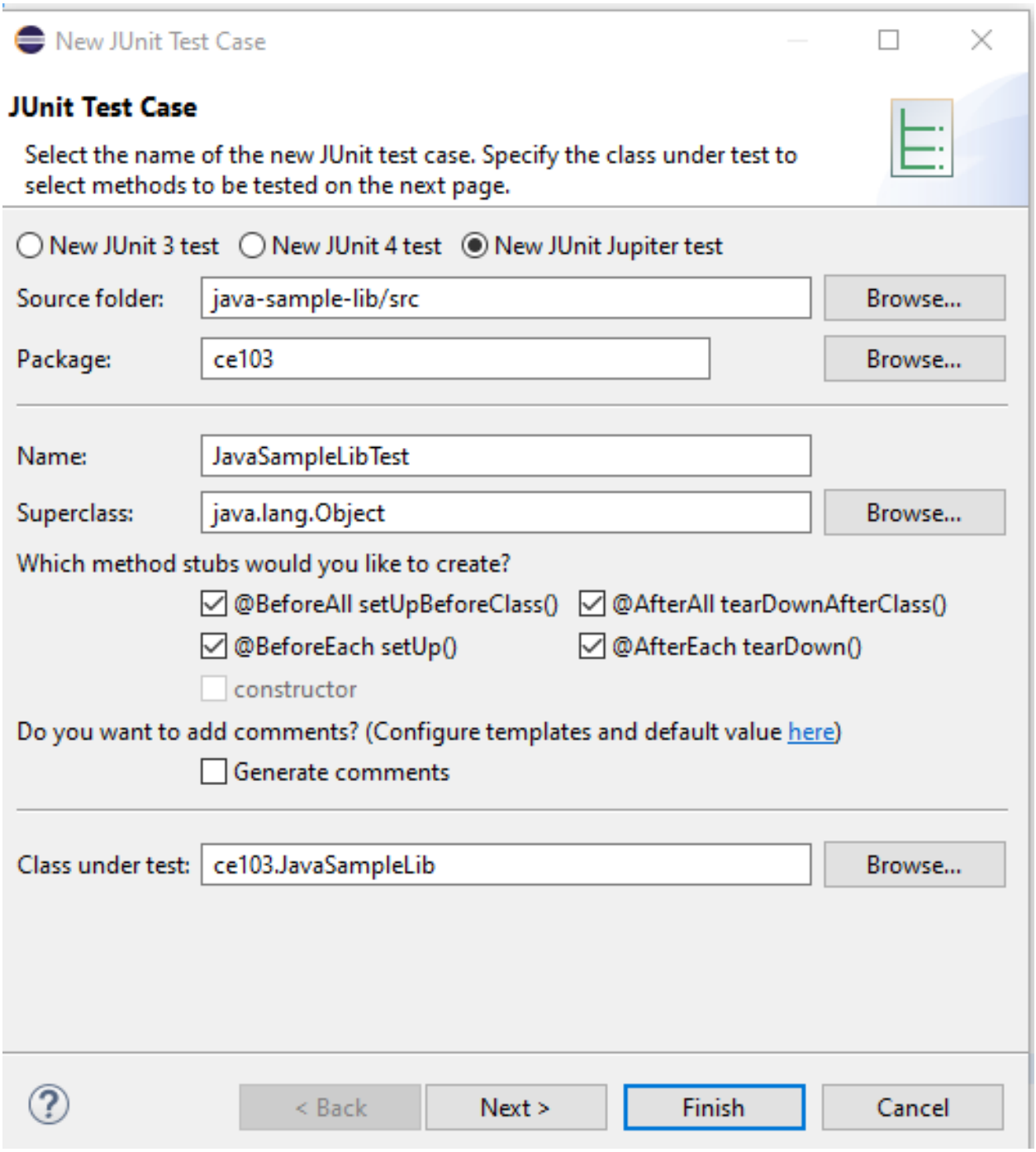
https://yasinmemic.medium.com/java-ile-unit-test-yazmak-birim-test-ca15cf0d024b

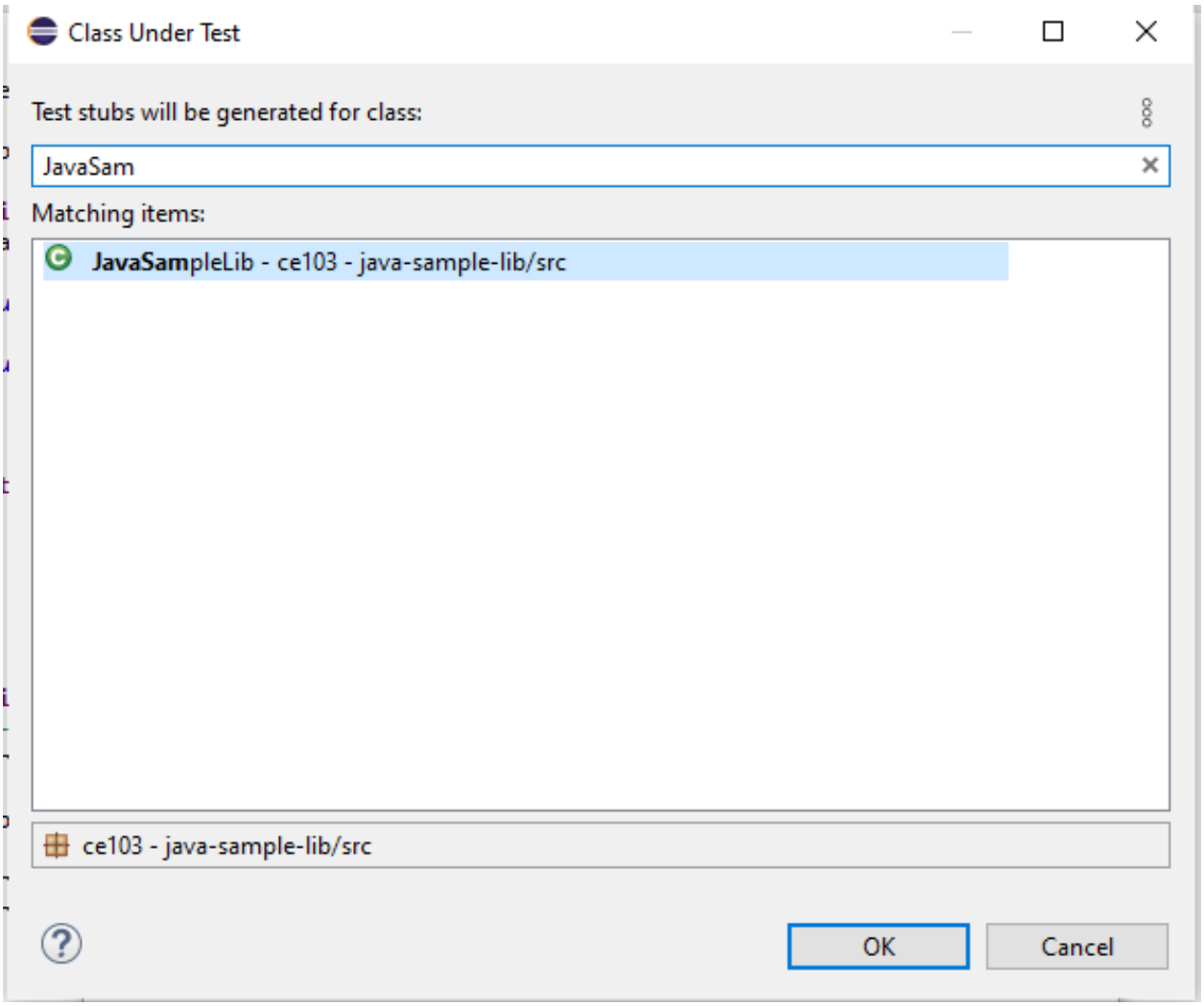
---

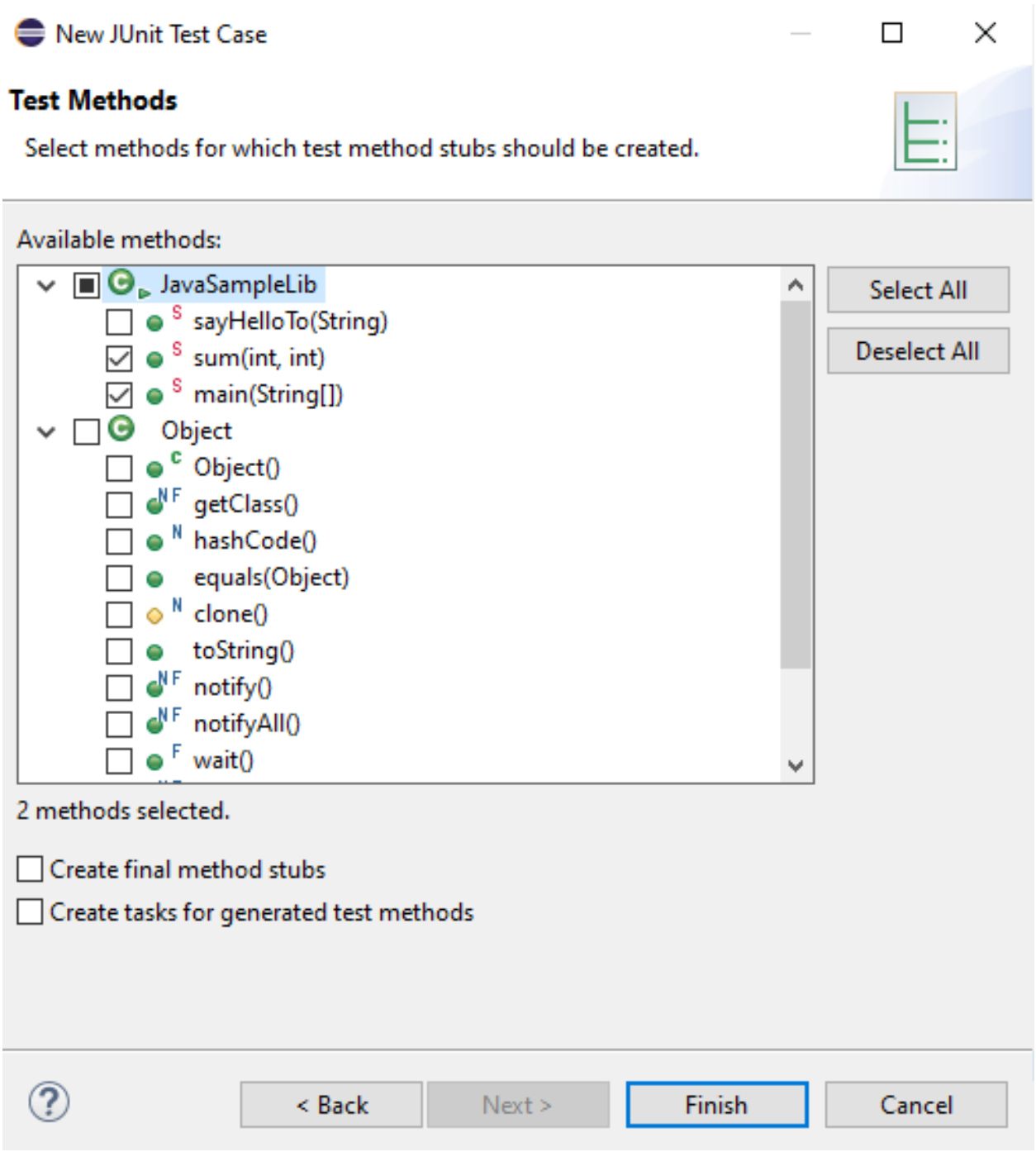
1.0.6.2 Java Application + JUnit In normal java application we can right click the project java-sample-lib and add Junit case

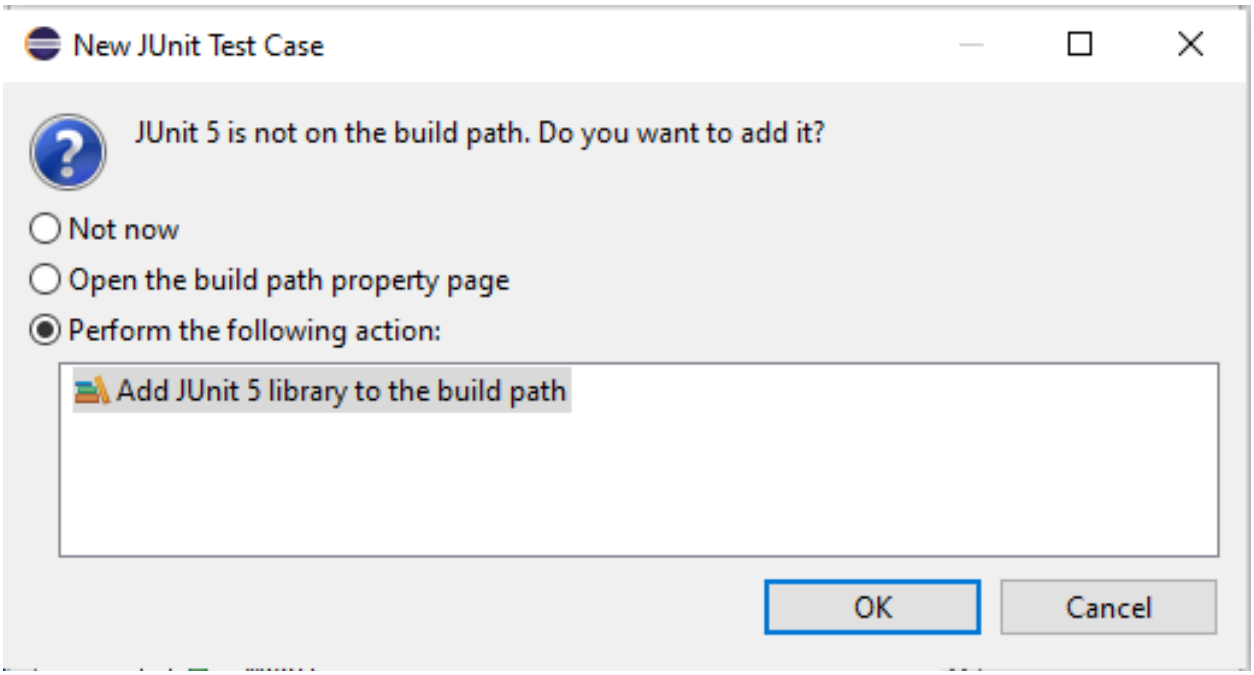
[22] http://codehelper.net/unit-testing/opencover-and-reportgenerator-unit-test-coverage-in-visual-studio-2013-and-2015/
[23] https://www.codeproject.com/Articles/1276980/OpenCover-and-ReportGenerator-Unit-Test-Coverage-i
[24] https://www.vogella.com/tutorials/JUnit/article.html
[25] https://junit.org/junit5/
[26] https://junit.org/junit5/docs/current/user-guide/
[27] https://examples.javacodegeeks.com/core-java/junit/junit-hello-world-example/


java-sample-app
java-sample-lib
JRE System Library [JavaSE-16]
src
ce103
New
Go Into
Open in New Window
Open Type Hierarchy F4
Show In Alt+Shift+W
Show in Local Terminal
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Build Path
Source Alt+Shift+S
Refactor Alt+Shift+T
Import...
Export...
References
Declarations
Java Project
Project...
Package
Class
Interface
Enum
Record
Annotation
Source Folder
Java Working Set
Folder
File
Untitled Text File
JUnit Test Case
Example...
Other... Ctrl+N
Create a JUnit test case
package ce103;
import java.io.IOExc
public class JavaSam
lic static vo
if(name.isBl
{
System.o
}else {
System.o
}
lic static in
int c = 0;
c = a+b;
return c;
lic static vo
// TODO Auto
System.out.p
JavaSampleLi
int result =
System.out.p
32
try {


New JUnit Test Case
JUnit Test Case
Select the name of the new JUnit test case. Specify the class under test to select methods to be tested on the next page.
New JUnit 3 test
New JUnit 4 test
New JUnit Jupiter test
Source folder:
java-sample-lib/src
Browse...
Package:
ce103
Browse...
Name:
JavaSampleLibTest
Superclass:
java.lang.Object
Browse...
Which method stubs would you like to create?
@BeforeAll setUpBeforeClass()
@AfterAll tearDownAfterClass()
@BeforeEach setUp()
@AfterEach tearDown()
constructor
Do you want to add comments? (Configure templates and default value here)
Generate comments
Class under test:
ce103.JavaSampleLib
Browse...
< Back
Next >
Finish
Cancel

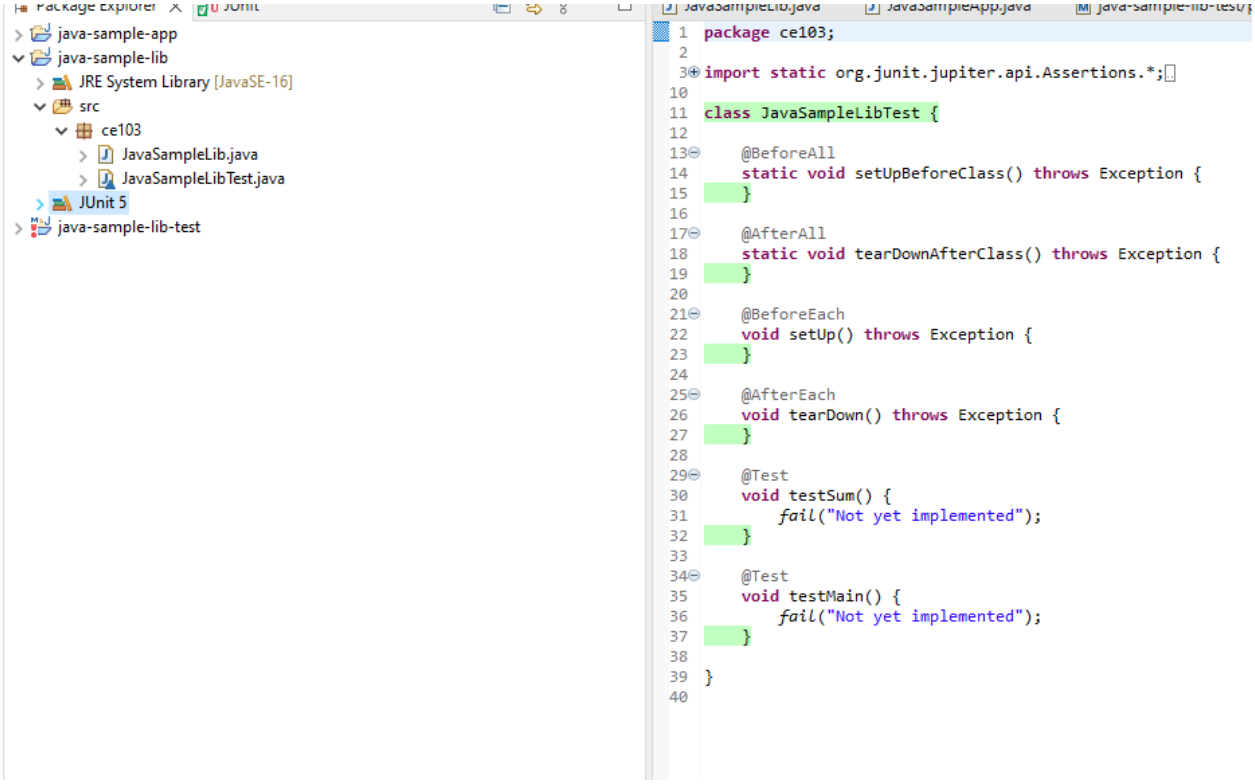

Class Under Test
Test stubs will be generated for class:
JavaSam
Matching items:
JavaSampleLib - ce103 - java-sample-lib/src
ce103 - java-sample-lib/src
OK
Cancel


New JUnit Test Case
Test Methods
Select methods for which test method stubs should be created.
Available methods:
JavaSampleLib
sayHelloTo(String)
sum(int, int)
main(String[])
Object
Object()
getClass()
hashCode()
equals(Object)
clone()
toString()
notify()
notifyAll()
wait()
Select All
Deselect All
2 methods selected.
Create final method stubs
Create tasks for generated test methods
< Back
Next >
Finish
Cancel



---

and you will have the following test class



Now we will create tests that check our function flowchart and return values

We need to cover all code branches that we coded

I have updated JavaSampleLib.java as follow to check outputs

JavaSampleLib.java

```java
package ce103;

public class JavaSampleLib {

    public static String sayHelloTo(String name) {

        String output = "";

        if(!name.isBlank() && !name.isEmpty()){
            output = "Hello "+name;
        }else {
            output = "Hello There";
        }

        System.out.println(output);

        return output;
    }

    public static int sum(int a,int b)
    {
        int c = 0;
        c = a+b;
        return c;
    }

    public int multiply(int a, int b) {
        return a * b;
    }

//    public static void main(String[] args) {
//        // TODO Auto-generated method stub
//        System.out.println("Hello World!");
//
//        JavaSampleLib.sayHelloTo("Computer");
//        int result = JavaSampleLib.sum(5, 4);
//        System.out.println("Results is" + result);
//        System.out.printf("Results is %d \n", result);
//
//
//        try {
//            System.in.read();
//        } catch (IOException e) {
//            // TODO Auto-generated catch block
//            e.printStackTrace();
//        }
//
//    }

}
```

---

and JavaSampleLibTest.java

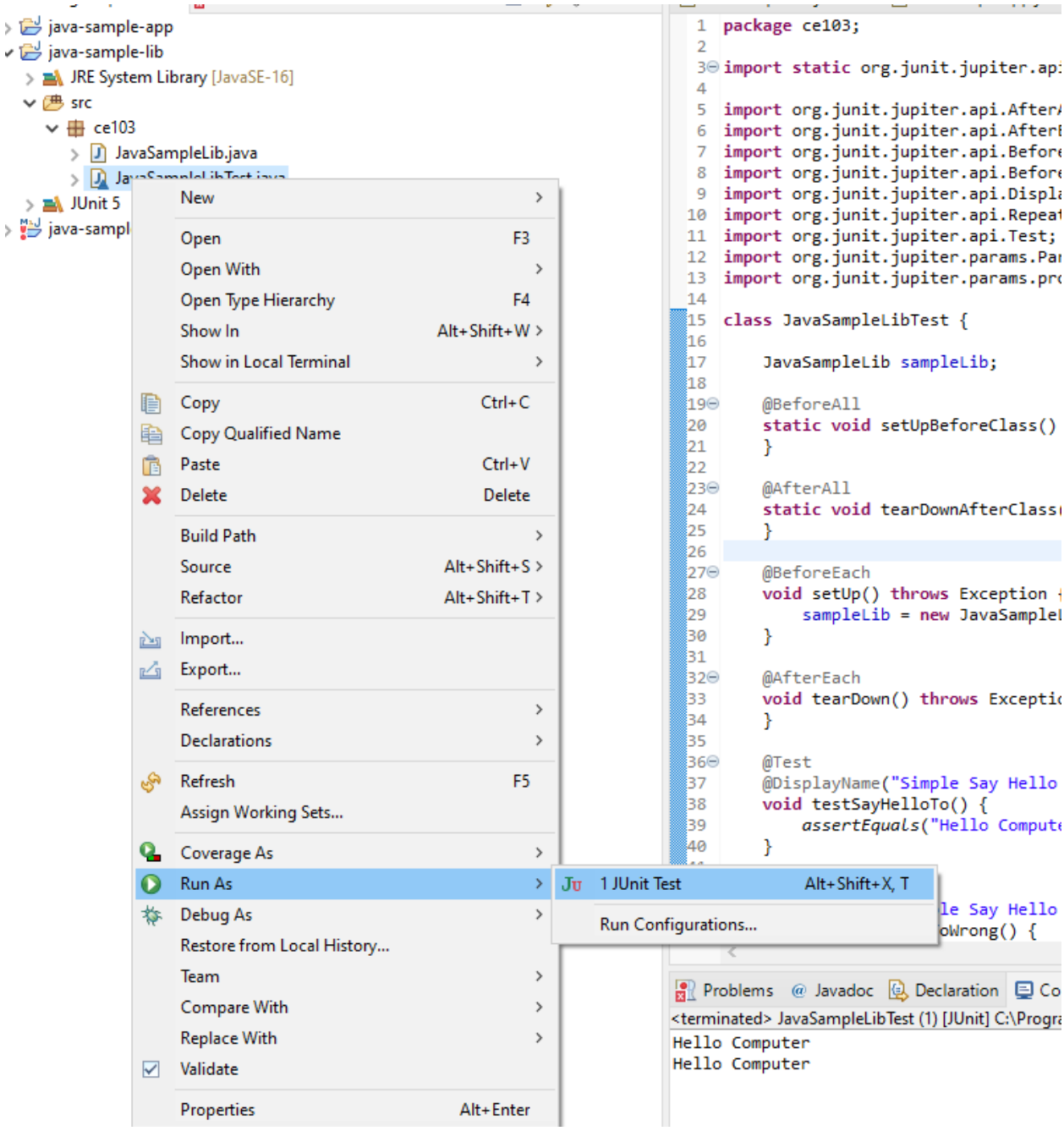
```java
package ce103;

import static org.junit.jupiter.api.Assertions.*;
```

```java
import org.junit.jupiter.api.AfterAll;
import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.BeforeAll;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.DisplayName;
import org.junit.jupiter.api.RepeatedTest;
import org.junit.jupiter.api.Test;
import org.junit.jupiter.params.ParameterizedTest;
import org.junit.jupiter.params.provider.MethodSource;

class JavaSampleLibTest {

    JavaSampleLib sampleLib;

    @BeforeAll
    static void setUpBeforeClass() throws Exception {
    }

    @AfterAll
    static void tearDownAfterClass() throws Exception {
    }

    @BeforeEach
    void setUp() throws Exception {
        sampleLib = new JavaSampleLib();
    }

    @AfterEach
    void tearDown() throws Exception {
    }

    @Test
    @DisplayName("Simple Say Hello should work")
    void testSayHelloTo() {
        assertEquals("Hello Computer", JavaSampleLib.sayHelloTo("Computer"), "Regular say hello should w
    }

    @Test
    @DisplayName("Simple Say Hello shouldn' work")
    void testSayHelloToWrong() {
        assertEquals("Hello All", JavaSampleLib.sayHelloTo("Computer"), "Regular say hello won't work")
    }


    @Test
    @DisplayName("Simple sum should work")
    void testSumCorrect() {
        assertEquals(9, JavaSampleLib.sum(4, 5), "Regular sum should work");
    }

    @Test
    @DisplayName("Simple sum shouldn't work")
    void testSumWrong() {
        assertEquals(10, JavaSampleLib.sum(4, 5), "Regular sum shouldn't work");
    }
```

```
    @Test
    @DisplayName("Simple multiplication should work")
    void testMultiply() {
        assertEquals(20, sampleLib.multiply(4, 5), "Regular multiplication should work");
    }

    @RepeatedTest(5)
    @DisplayName("Ensure correct handling of zero")
    void testMultiplyWithZero() {
        assertEquals(0, sampleLib.multiply(0, 5), "Multiple with zero should be zero");
        assertEquals(0, sampleLib.multiply(5, 0), "Multiple with zero should be zero");
    }

    public static int[][] data() {
        return new int[][] { { 1, 2, 2 }, { 5, 3, 15 }, { 121, 4, 484 },{ 2, 2, 2 } };
    }

    @ParameterizedTest
    @MethodSource(value = "data")
    void testWithStringParameter(int[] data) {
        JavaSampleLib tester = new JavaSampleLib();
        int m1 = data[0];
        int m2 = data[1];
        int expected = data[2];
        assertEquals(expected, tester.multiply(m1, m2));
    }

}
```

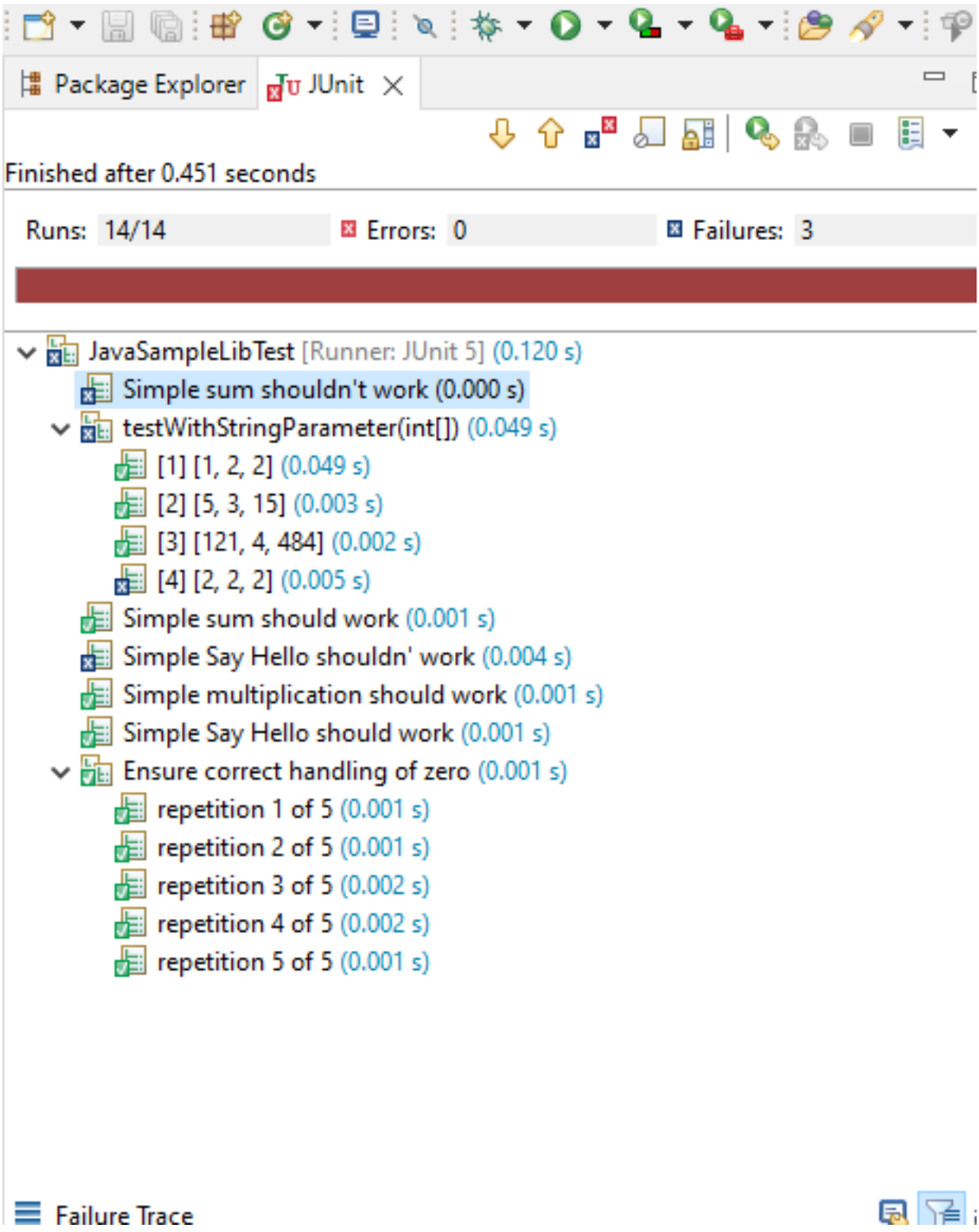
---

if we run tests



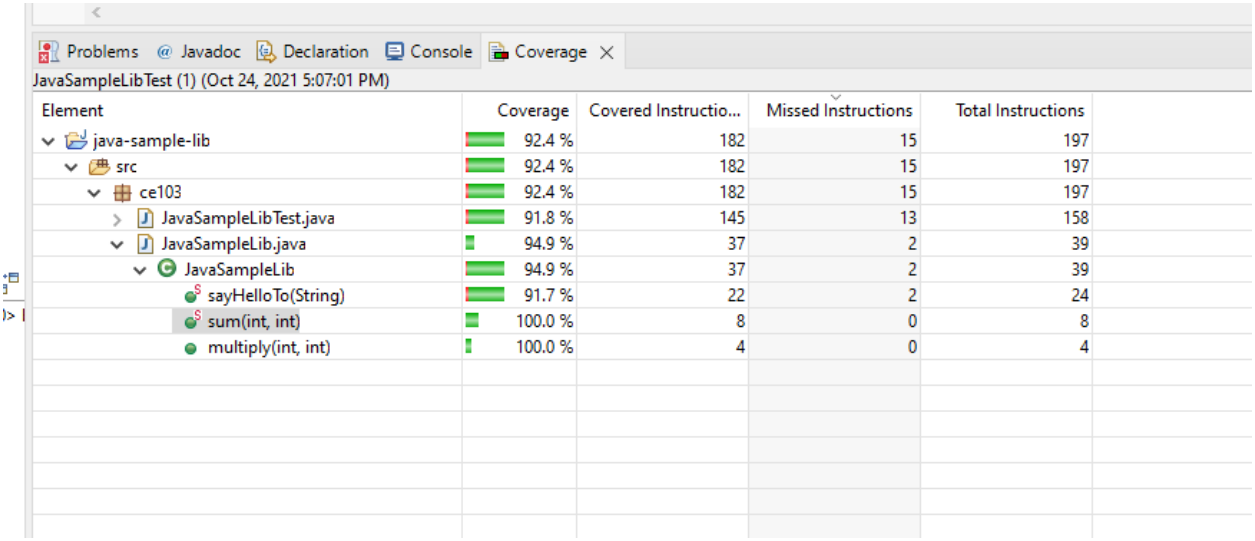
---

we will see all results there



---

also we can see the code coverage of tests



---

when we open our source code (just close and open again another case highlighting will not work) you will see tested part of your codes

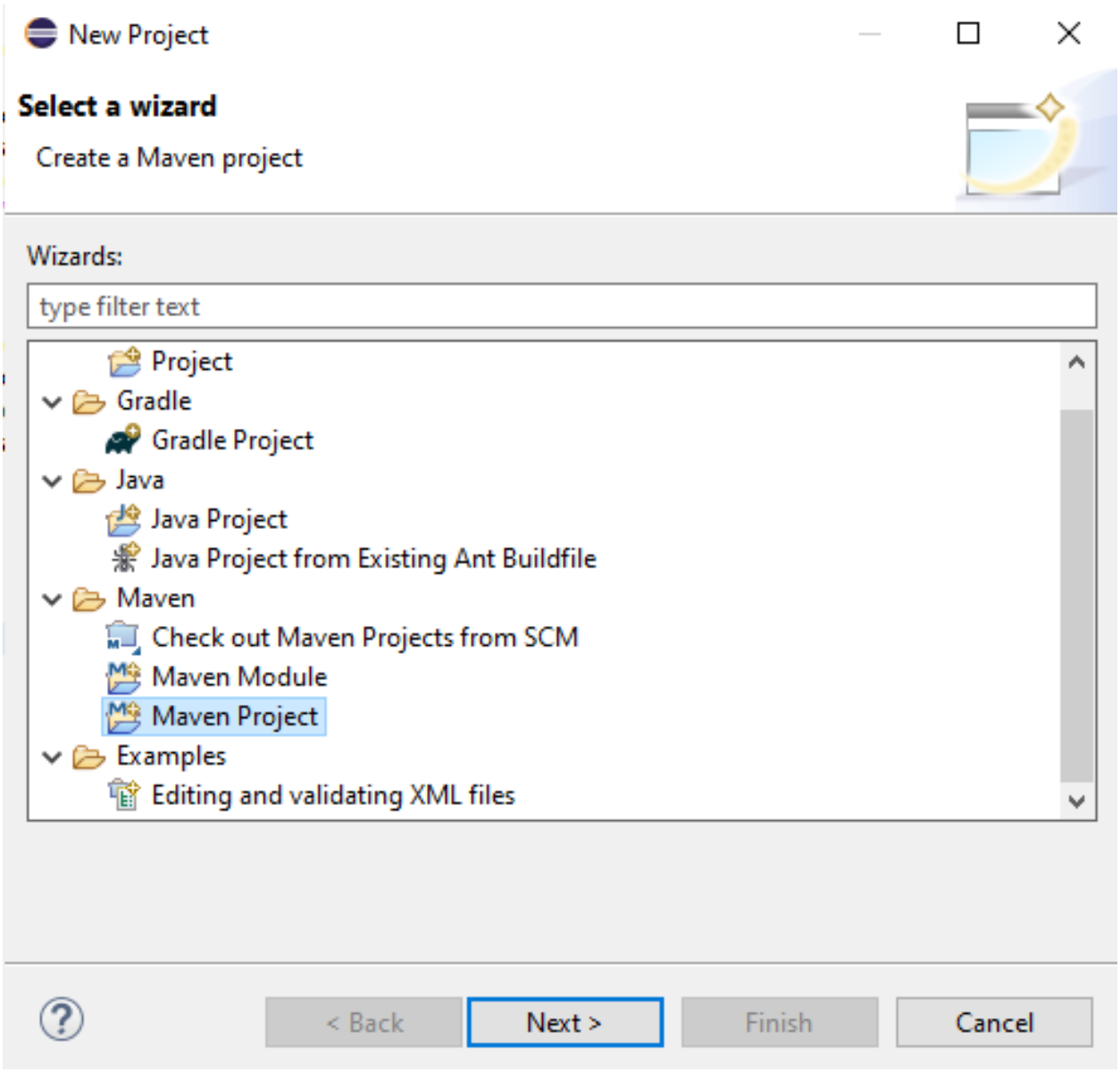
```java
package ce103;

public class JavaSampleLib {

    public static String sayHelloTo(String name) {

        String output = "";

        if(!name.isBlank() && !name.isEmpty()){
            output = "Hello "+name;
        }else {
            output = "Hello There";
        }

        System.out.println(output);

        return output;
    }

    public static int sum(int a,int b)
    {
        int c = 0;
        c = a+b;
        return c;
    }

    public int multiply(int a, int b) {
        return a * b;
    }

```

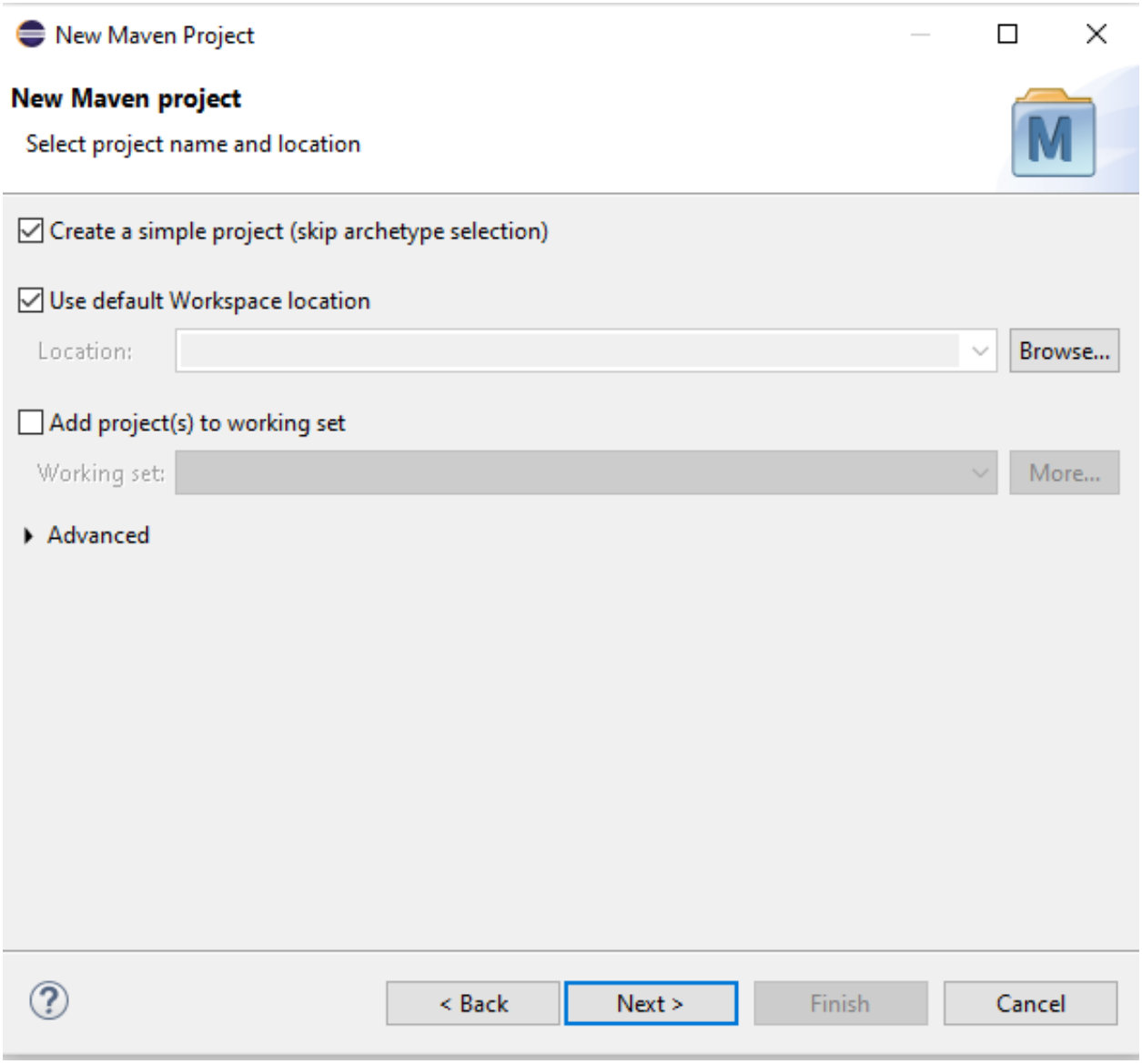
1.0.6.3 Maven Java Application + JUnit

Lets create Maven project with tests

Create a maven project

File -> New -> Maven Project


New Project
Select a wizard
Create a Maven project
Wizards:
type filter text
Project
Gradle
Gradle Project
Java
Java Project
Java Project from Existing Ant Buildfile
Maven
Check out Maven Projects from SCM
Maven Module
Maven Project
Examples
Editing and validating XML files
< Back
Next >
Finish
Cancel



Lets convert our sample java-sample-lib directories to standard folder structure for test and app division

Maven – Introduction to the Standard Directory Layout[28]

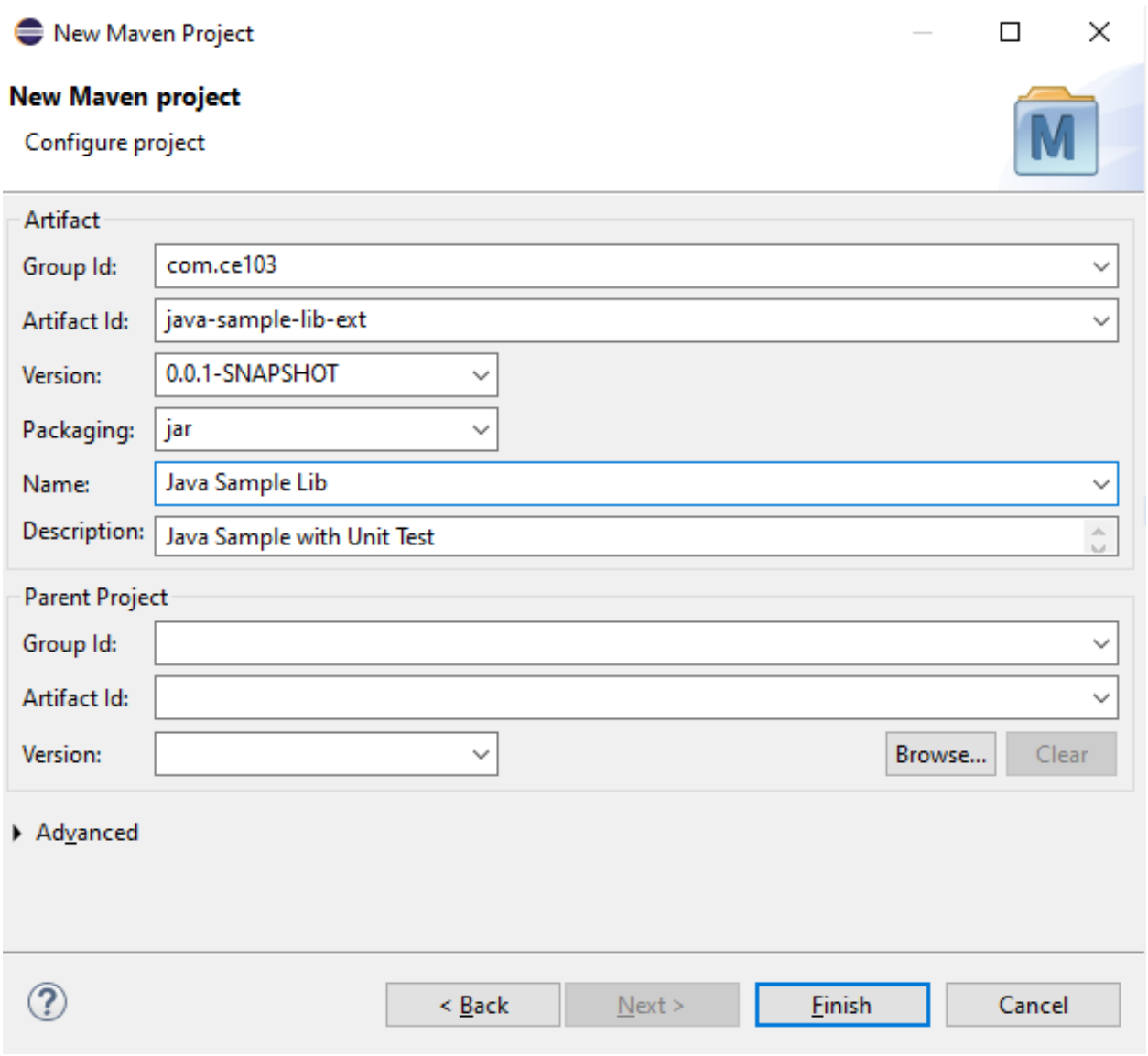
Also for intro you can use this

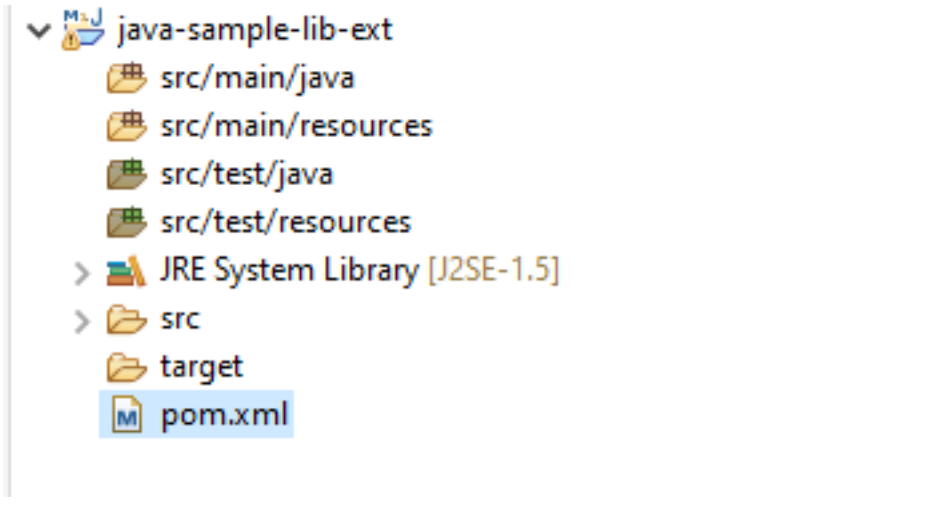
JUnit Hello World Example - Examples Java Code Geeks - 2021[29]

Eclipse
Maven
Java
JUnit 4.12 (pulled by Maven automatically)

Lets give new sample java-sample-lib-mvnbut in this time we will create a maven project

[28] http://maven.apache.org/guides/introduction/introduction-to-the-standard-directory-layout.html
[29] https://examples.javacodegeeks.com/core-java/junit/junit-hello-world-example/


New Maven Project
New Maven project
Configure project
Artifact
Group Id:
com.ce103
Artifact Id:
java-sample-lib-ext
Version:
0.0.1-SNAPSHOT
Packaging:
jar
Name:
Java Sample Lib
Description:
Java Sample with Unit Test
Parent Project
Group Id:
Artifact Id:
Version:
Browse...
Clear
Advanced
< Back
Next >
Finish
Cancel


java-sample-lib-ext
src/main/java
src/main/resources
src/test/java
src/test/resources
JRE System Library [J2SE-1.5]
src
target
pom.xml

pom.xml file

```
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.ce103</groupId>
  <artifactId>java-sample-lib-ext</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <name>Java Sample Lib</name>
  <description>Java Sample with Unit Test</description>
</project>
```

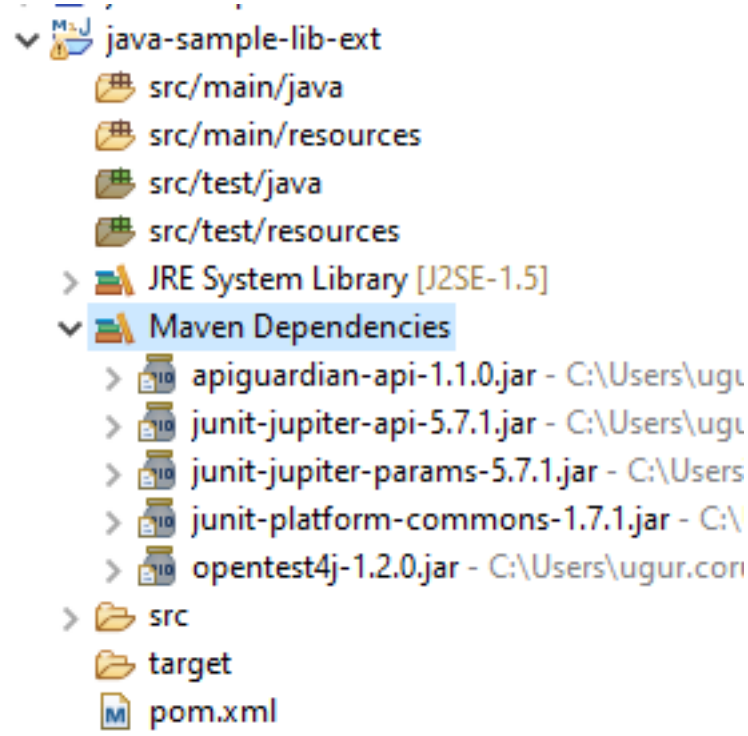
---

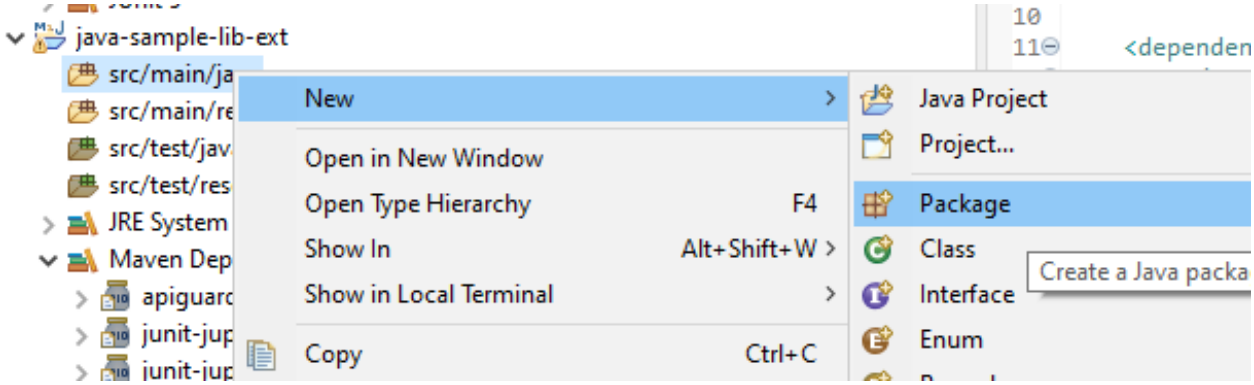
we will add JUnit 5 for our project

```
<project xmlns="http://maven.apache.org/POM/4.0.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd"
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.ce103</groupId>
    <artifactId>java-sample-lib-ext</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>Java Sample Lib</name>
    <description>Java Sample with Unit Test</description>

    <dependencies>
        <dependency>
            <groupId>org.junit.jupiter</groupId>
            <artifactId>junit-jupiter-params</artifactId>
            <version>5.7.1</version>
            <scope>test</scope>
        </dependency>
    </dependencies>

</project>
```

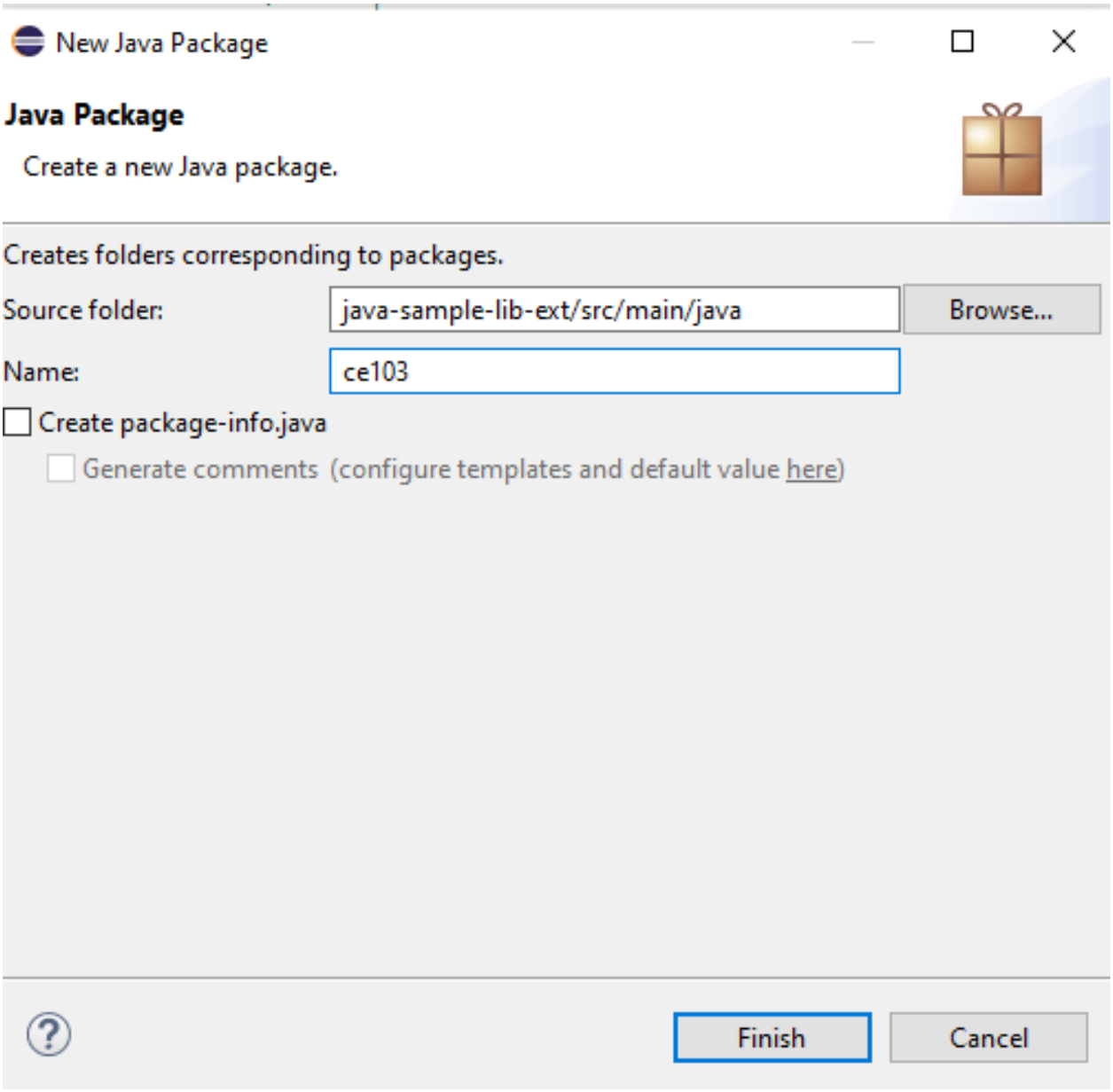
---

it will automatically download libraries

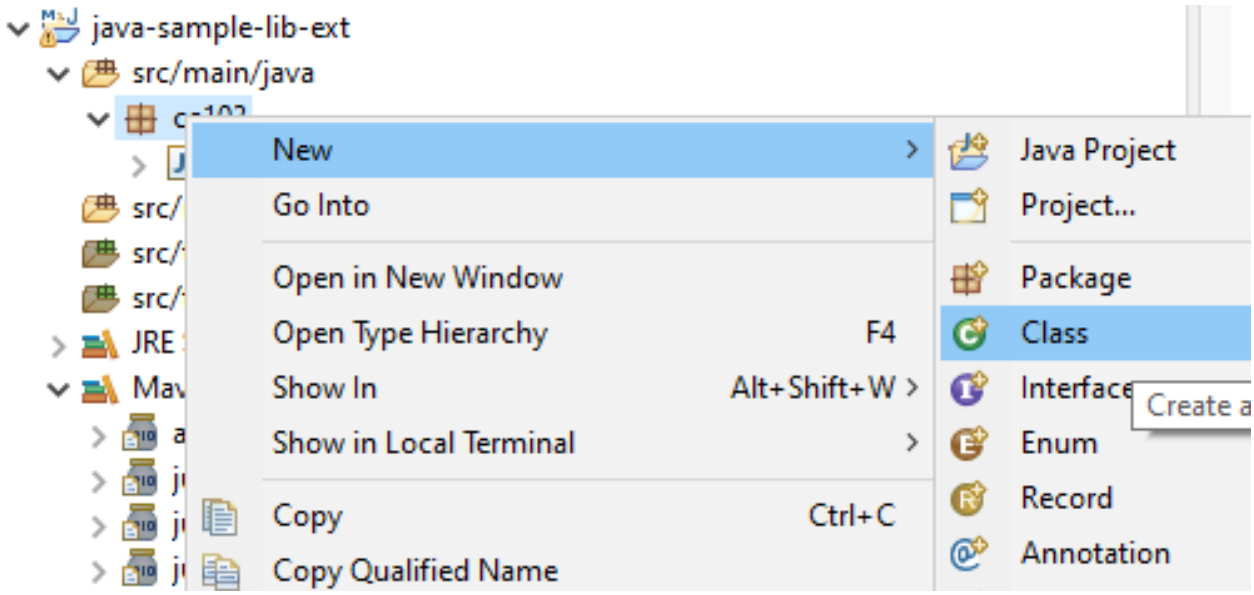


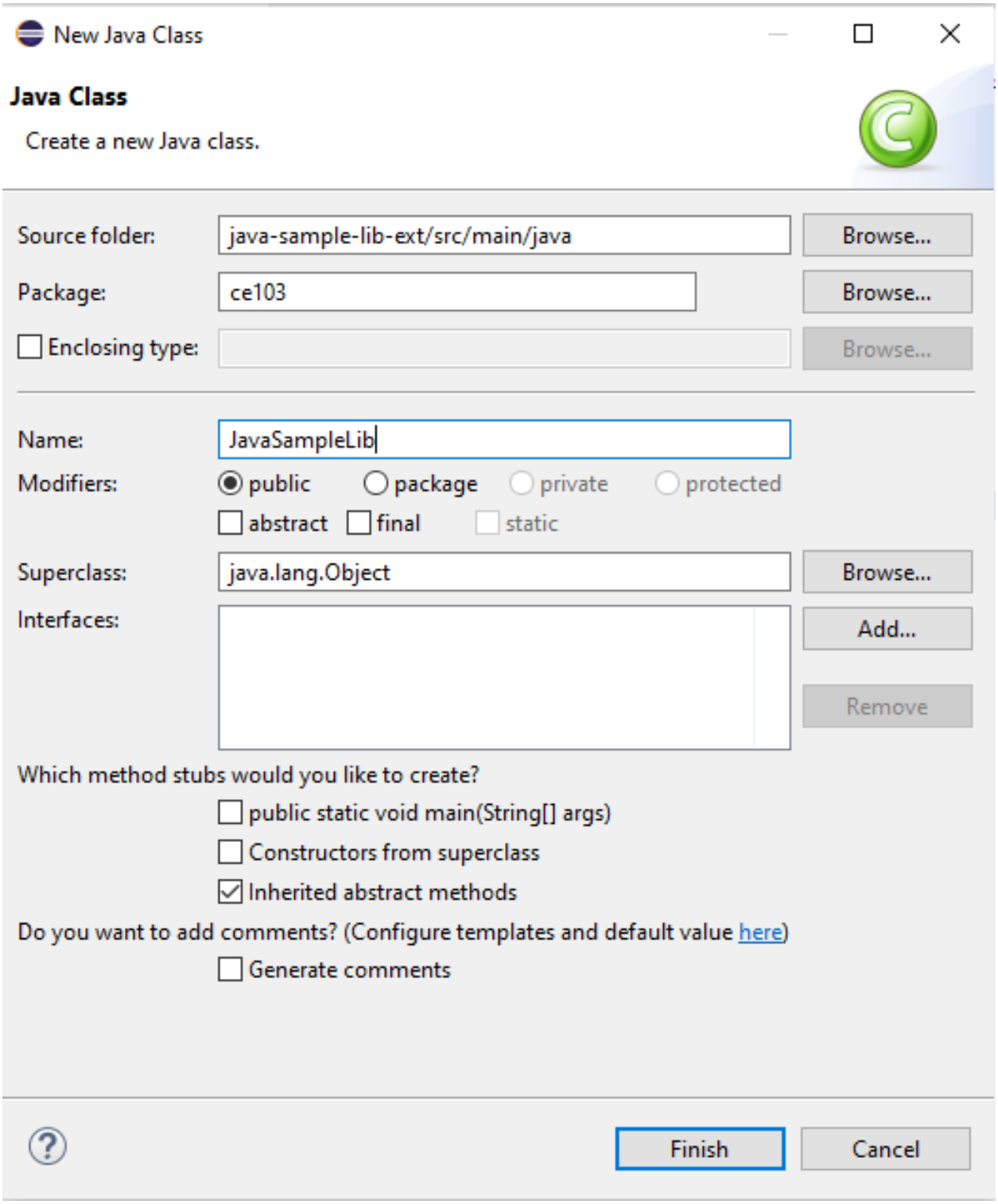


Create java sample library in ce103 package, first create java package



In this package create library class


java-sample-lib-ext
src/main/java
New
Go Into
Open in New Window
Open Type Hierarchy F4
Show In Alt+Shift+W
Show in Local Terminal
Copy Ctrl+C
Copy Qualified Name
Java Project
Project...
Package
Class
Interface
Enum
Record
Annotation



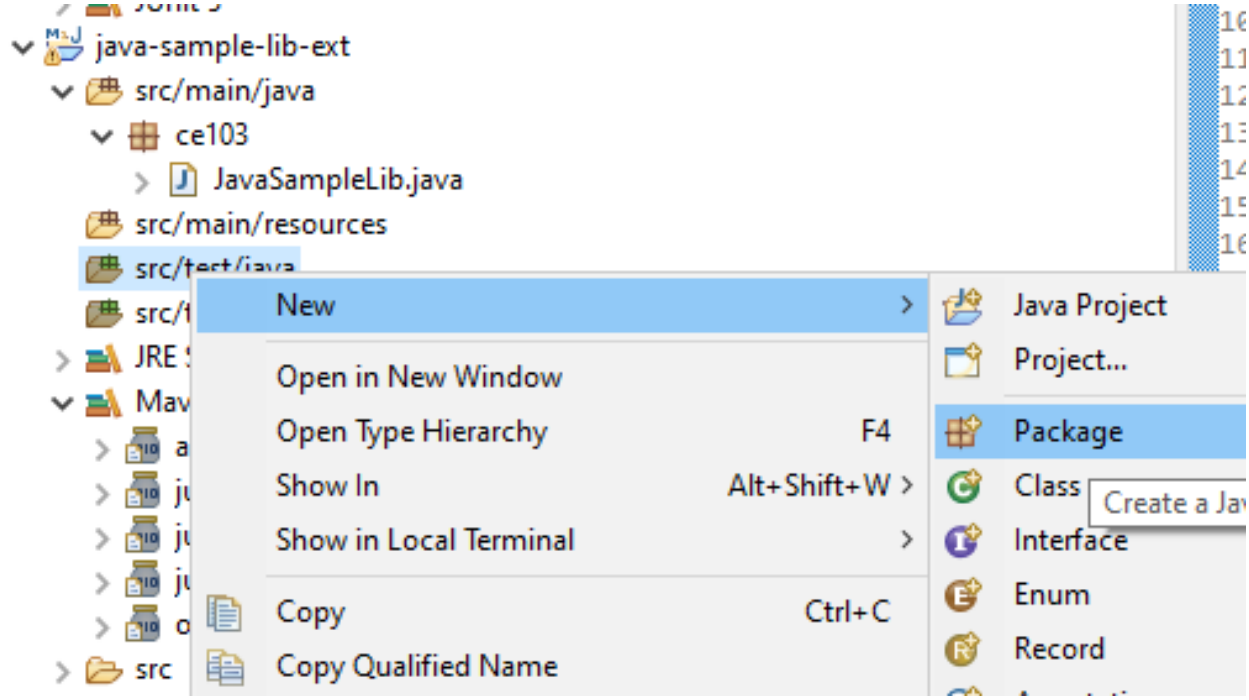
---

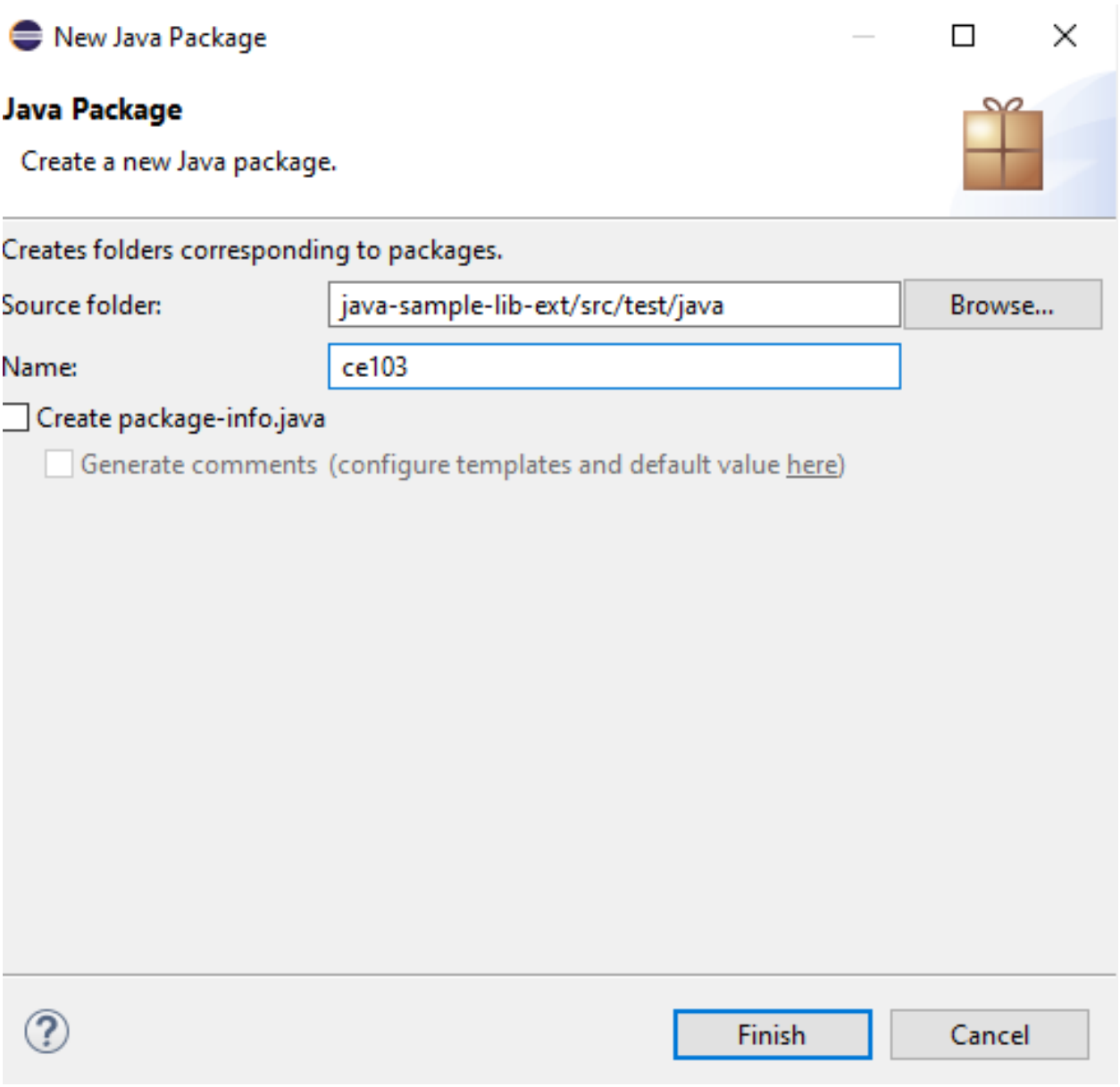
copy content from other library

```
package ce103;

public class JavaSampleLib {

public static String sayHelloTo(String name) {
```

```
        String output = "";

        if(!name.isBlank() && !name.isEmpty()){
            output = "Hello "+name;
        }else {
            output = "Hello There";
        }

        System.out.println(output);

        return output;
    }

    public static int sum(int a,int b)
    {
        int c = 0;
        c = a+b;
        return c;
    }

    public int multiply(int a, int b) {
        return a * b;
    }


}
```

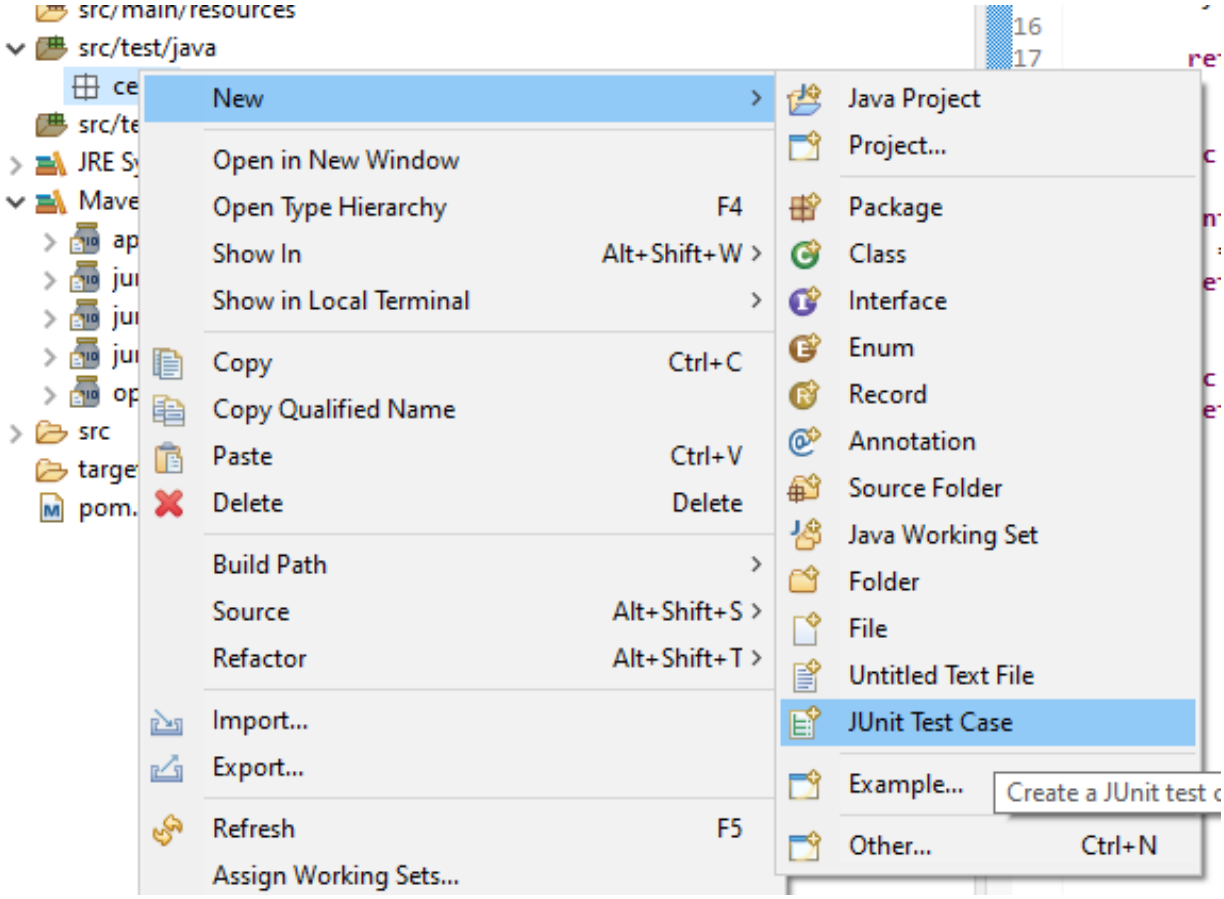
Now lets create tests inf src/test/java

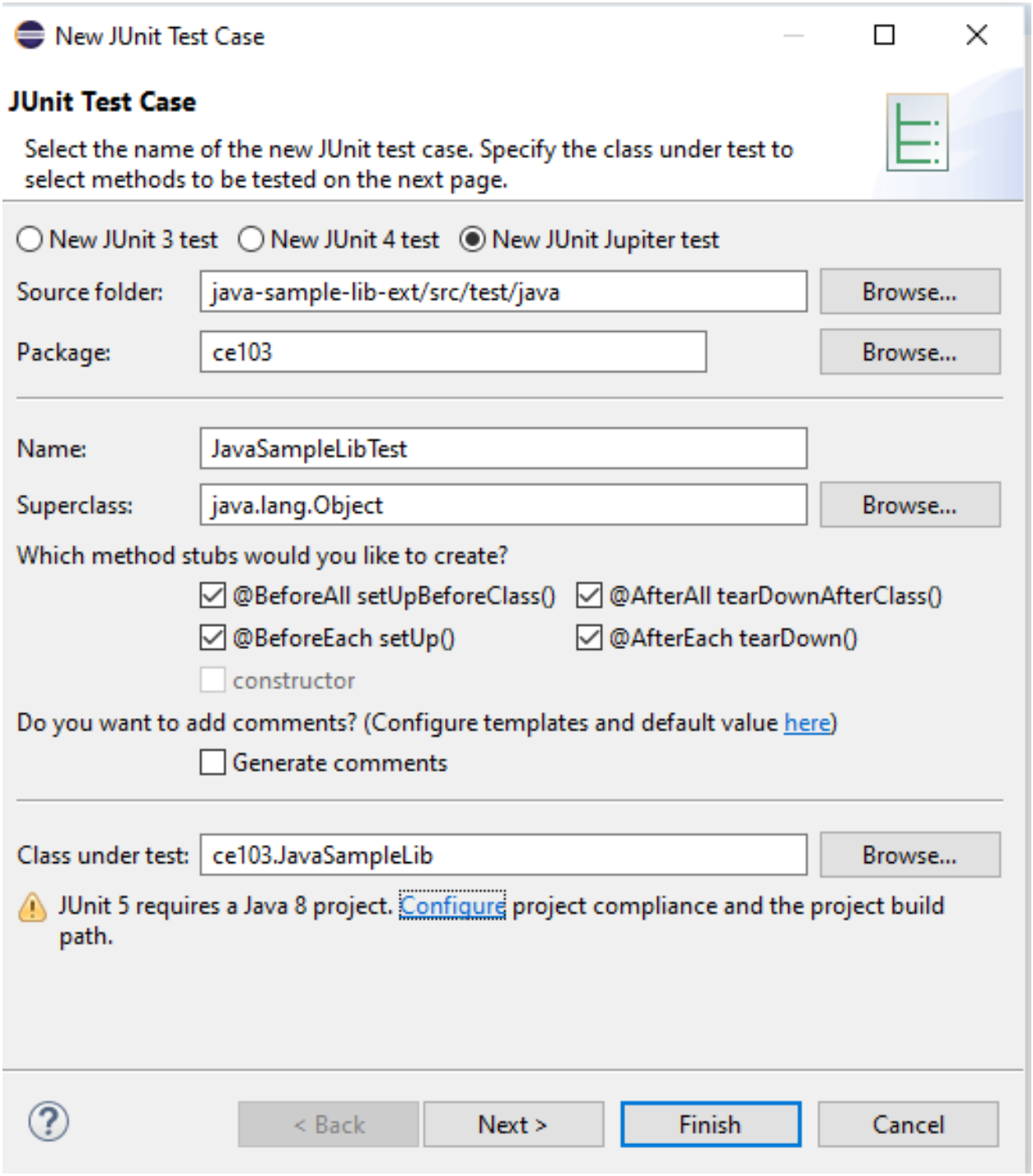


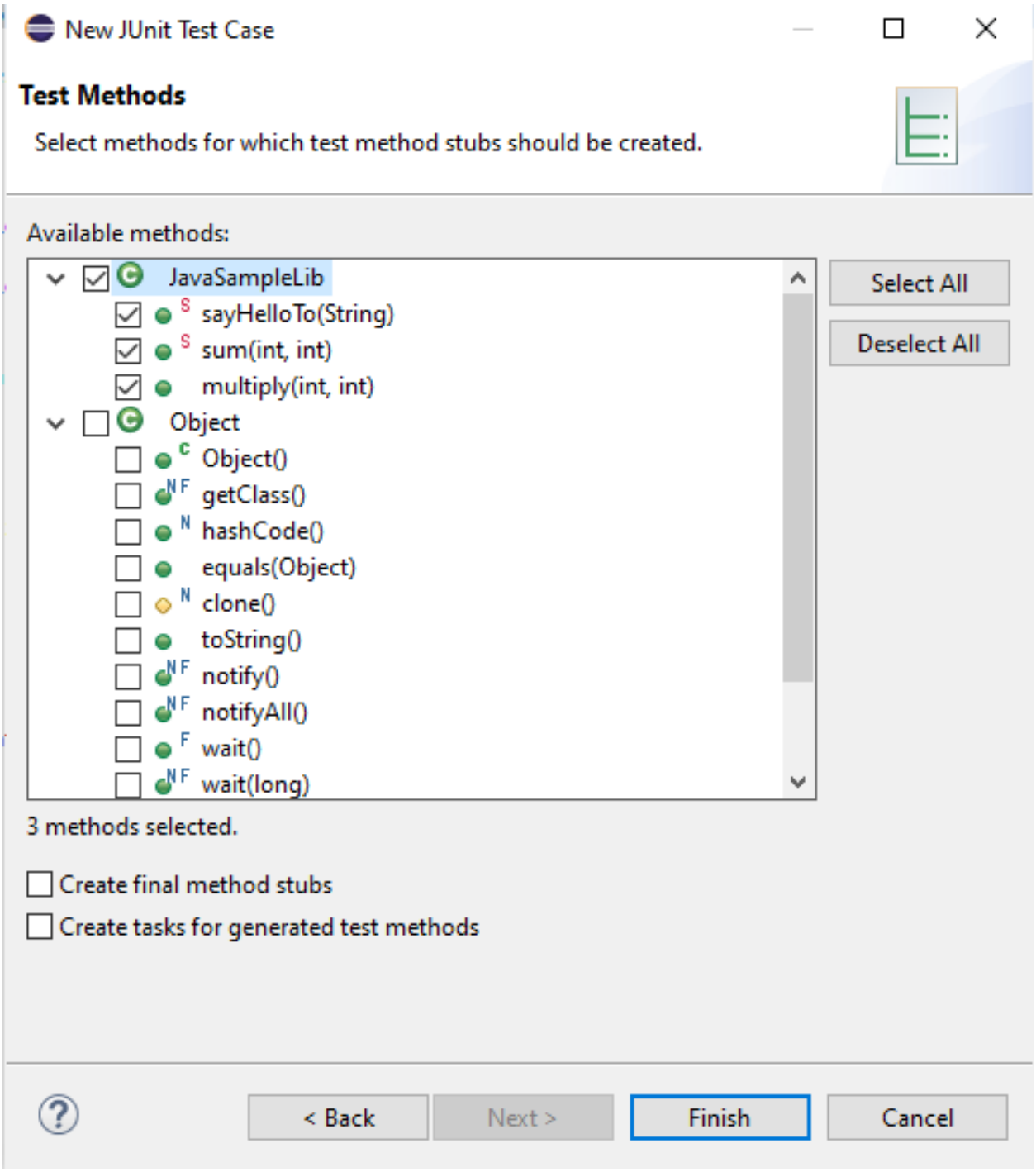


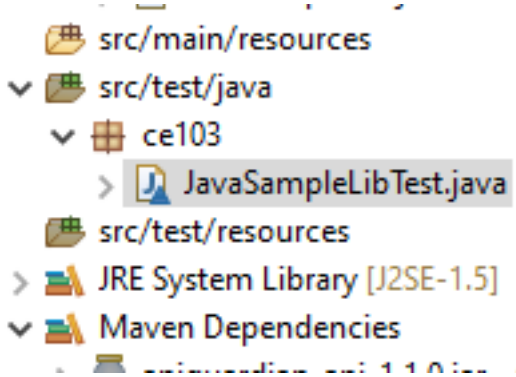
---

create a JUnit Case


src/main/resources
src/test/java
New
Open in New Window
Open Type Hierarchy F4
Show In Alt+Shift+W
Show in Local Terminal
Copy Ctrl+C
Copy Qualified Name
Paste Ctrl+V
Delete Delete
Build Path
Source Alt+Shift+S
Refactor Alt+Shift+T
Import...
Export...
Refresh F5
Assign Working Sets...
Java Project
Project...
Package
Class
Interface
Enum
Record
Annotation
Source Folder
Java Working Set
Folder
File
Untitled Text File
JUnit Test Case
Example...
Other... Ctrl+N
Create a JUnit test


New JUnit Test Case
JUnit Test Case
Select the name of the new JUnit test case. Specify the class under test to select methods to be tested on the next page.
New JUnit 3 test
New JUnit 4 test
New JUnit Jupiter test
Source folder:
java-sample-lib-ext/src/test/java
Browse...
Package:
ce103
Browse...
Name:
JavaSampleLibTest
Superclass:
java.lang.Object
Browse...
Which method stubs would you like to create?
@BeforeAll setUpBeforeClass()
@AfterAll tearDownAfterClass()
@BeforeEach setUp()
@AfterEach tearDown()
constructor
Do you want to add comments? (Configure templates and default value here)
Generate comments
Class under test:
ce103.JavaSampleLib
Browse...
JUnit 5 requires a Java 8 project. Configure project compliance and the project build path.
< Back
Next >
Finish
Cancel


New JUnit Test Case
Test Methods
Select methods for which test method stubs should be created.
Available methods:
JavaSampleLib
sayHelloTo(String)
sum(int, int)
multiply(int, int)
Object
Object()
getClass()
hashCode()
equals(Object)
clone()
toString()
notify()
notifyAll()
wait()
wait(long)
Select All
Deselect All
3 methods selected.
Create final method stubs
Create tasks for generated test methods
< Back
Next >
Finish
Cancel



---

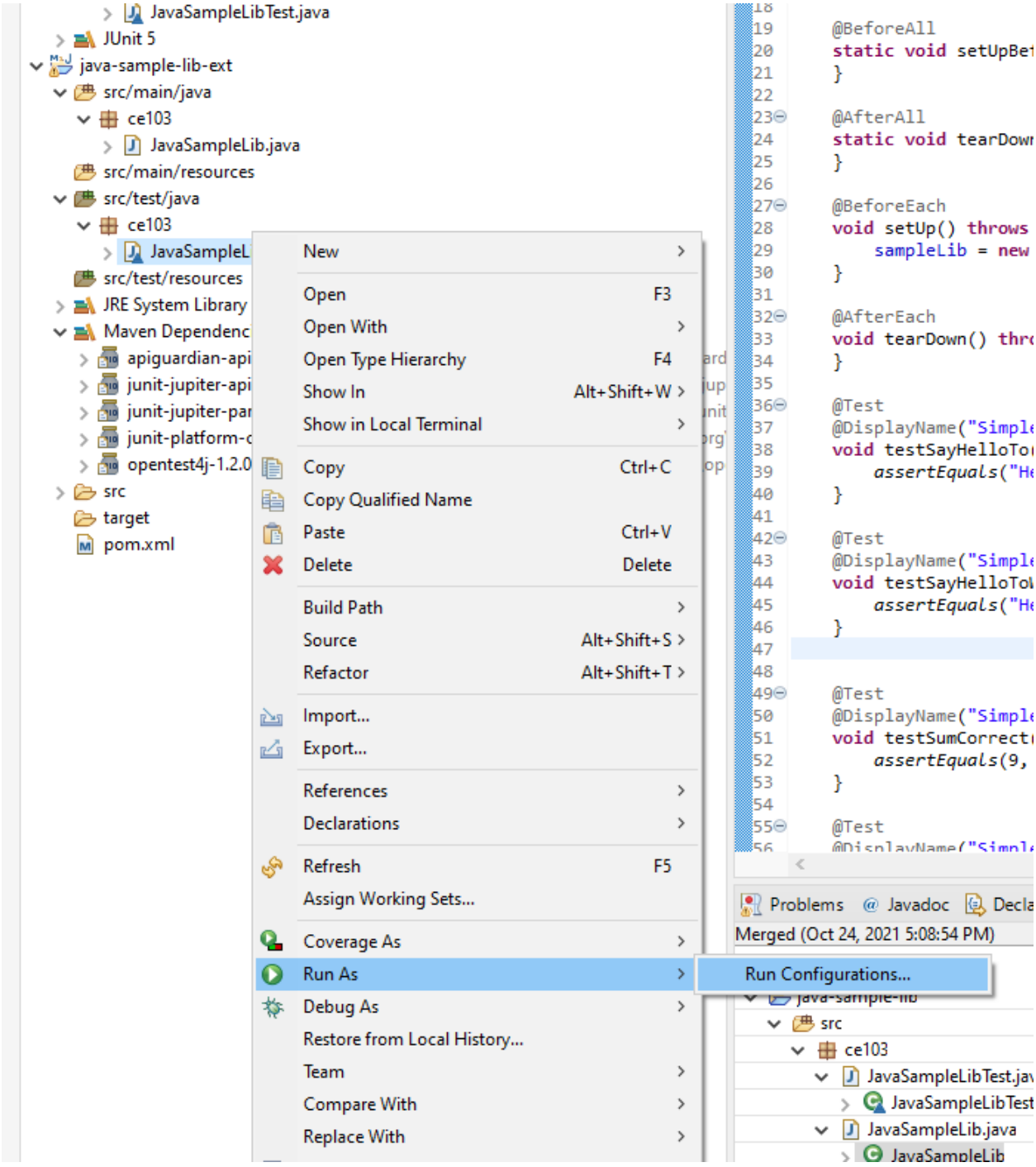
you will simple template

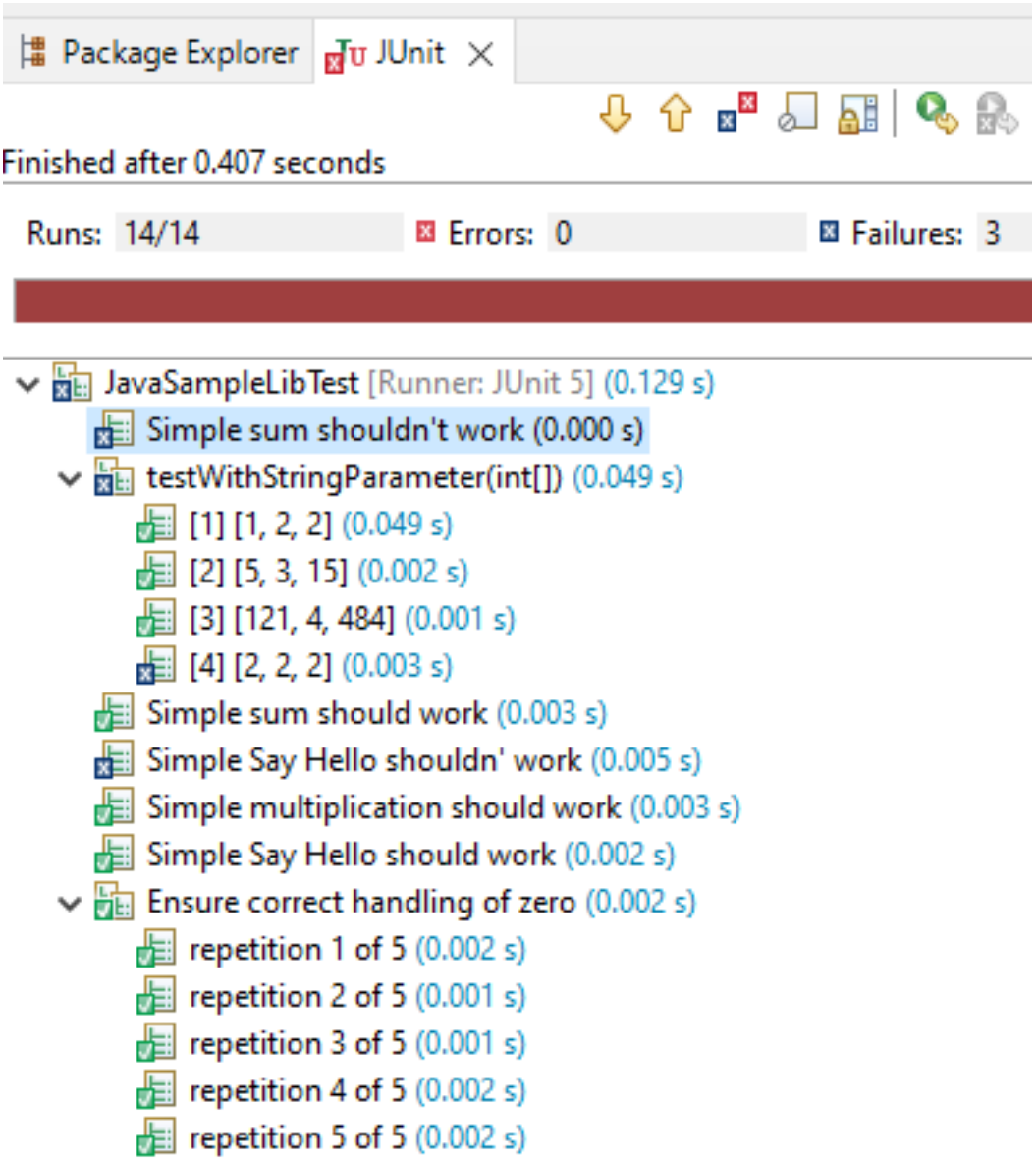
```
package ce103;

import static org.junit.jupiter.api.Assertions.*;

import org.junit.jupiter.api.AfterAll;
import org.junit.jupiter.api.AfterEach;
import org.junit.jupiter.api.BeforeAll;
import org.junit.jupiter.api.BeforeEach;
import org.junit.jupiter.api.Test;

class JavaSampleLibTest {

    @BeforeAll
    static void setUpBeforeClass() throws Exception {
    }

    @AfterAll
    static void tearDownAfterClass() throws Exception {
    }

    @BeforeEach
    void setUp() throws Exception {
    }

    @AfterEach
    void tearDown() throws Exception {
    }

    @Test
    void testSayHelloTo() {
        fail("Not yet implemented");
    }

    @Test
    void testSum() {
        fail("Not yet implemented");
    }

    @Test
    void testMultiply() {
        fail("Not yet implemented");
    }
```

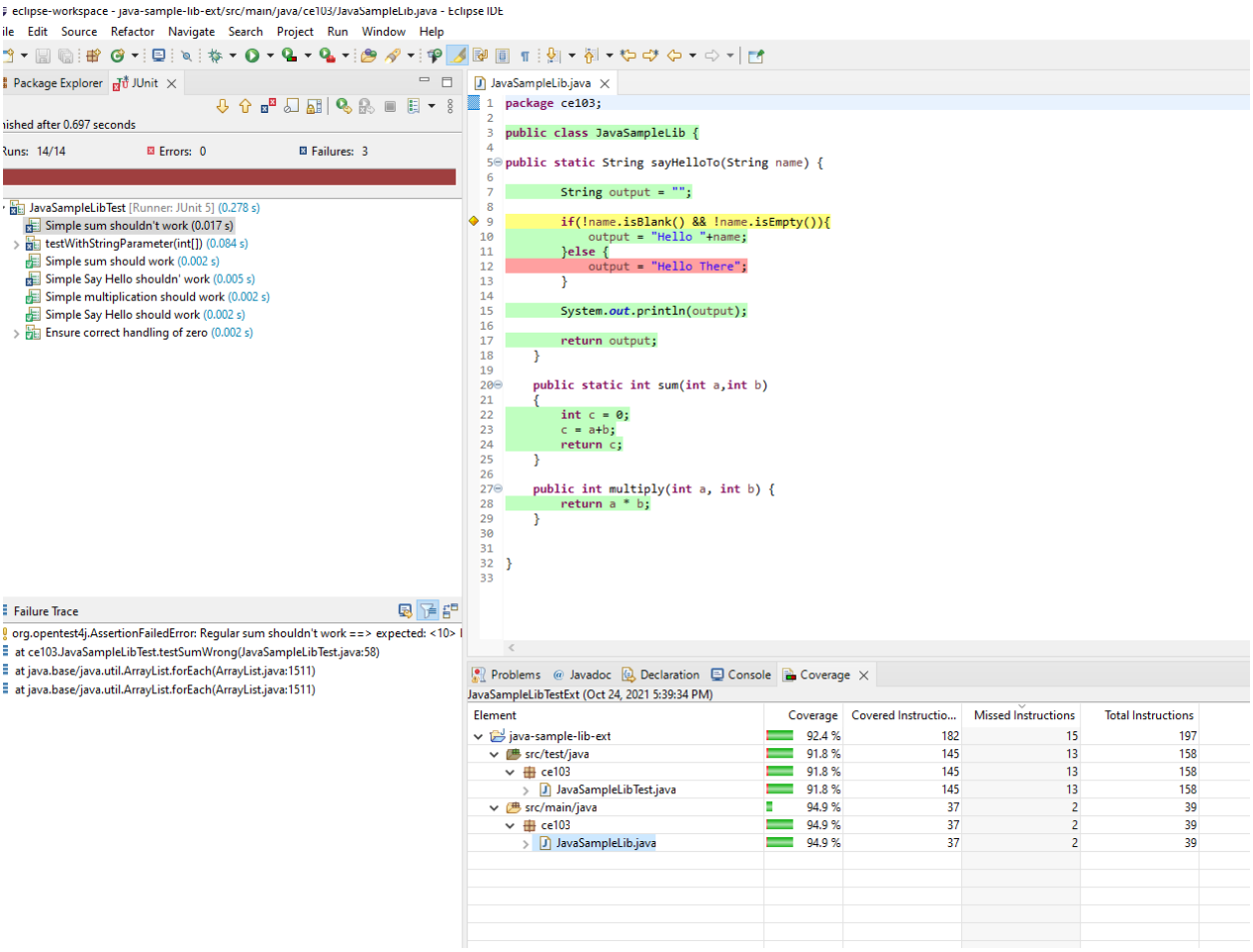
}

---

now lets copy tests from other projects




Package Explorer
JUnit
Finished after 0.407 seconds
Runs: 14/14
Errors: 0
Failures: 3
JavaSampleLibTest [Runner: JUnit 5] (0.129 s)
Simple sum shouldn't work (0.000 s)
testWithStringParameter(int[]) (0.049 s)
[1] [1, 2, 2] (0.049 s)
[2] [5, 3, 15] (0.002 s)
[3] [121, 4, 484] (0.001 s)
[4] [2, 2, 2] (0.003 s)
Simple sum should work (0.003 s)
Simple Say Hello shouldn' work (0.005 s)
Simple multiplication should work (0.003 s)
Simple Say Hello should work (0.002 s)
Ensure correct handling of zero (0.002 s)
repetition 1 of 5 (0.002 s)
repetition 2 of 5 (0.001 s)
repetition 3 of 5 (0.001 s)
repetition 4 of 5 (0.002 s)
repetition 5 of 5 (0.002 s)



That's a part of java unit testing...

1.1 TDD (Test Driven Development)

1.2 Test and Deployment Automation Management

1.2.0.1 Travis-CI + C

1.2.0.2 Travis-CI + Cpp

1.2.0.3 Travis-CI + C

1.2.0.4 Travis-CI + Java

2 References

GitHub - MicrosoftDocs/cpp-docs: C++ Documentation[30]

[30] https://github.com/MicrosoftDocs/cpp-docs