CE103 Algorithms and Programming I

Week-4

Introduction to Code Reusability and Automated Testing

Download DOC, SLIDE, PPTX

Outline

- Introduction to Code Reusability and Automated Testing
- Shared Library Development
  - C
  - C++
  - C#
  - Java
- Unit Testing
  - C
  - C++
  - C#
  - Java
- Continues Integration Platforms

Introduction to Code Reusability and Automated Testing

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

Selected Development Environment

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

Example Content

- Each example will include two function
- " `Hello <name>` " 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 the result to the sum variable.
- We will locate them in the library and use them from a console application, also we will create unit tests for testing their functionalities and return variables

Shared Library Development

C Programming (Static Library)

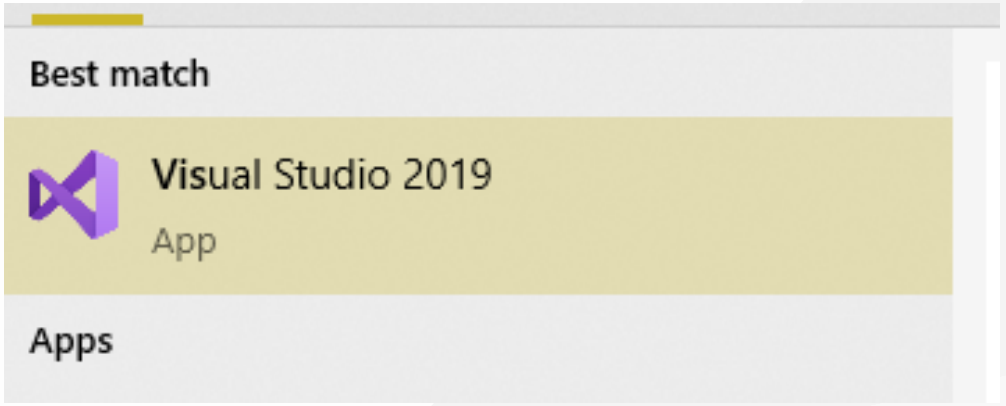
Visual Studio Community Edition

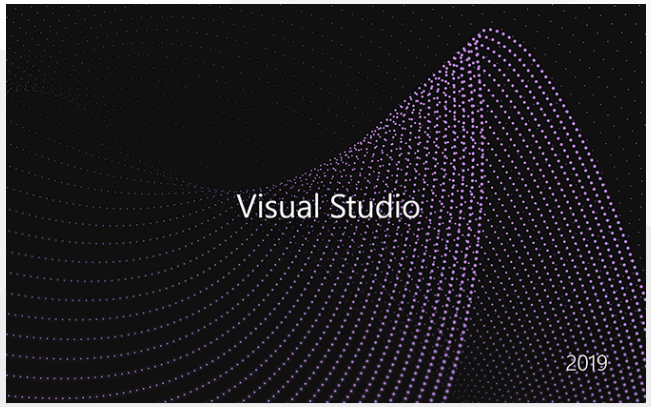
Shared Library Development - (VS C Static Library)-1

- In this sample, we will create a **c-lib-sample** project that contains a library, executable, unit tests and unit test runners.
- First of all, you install Visual Studio Community Edition from the website
  - Visual Studio 2019 Community Edition - Son Ücretsiz Sürümü İndir

Shared Library Development - (VS C Static Library)-2

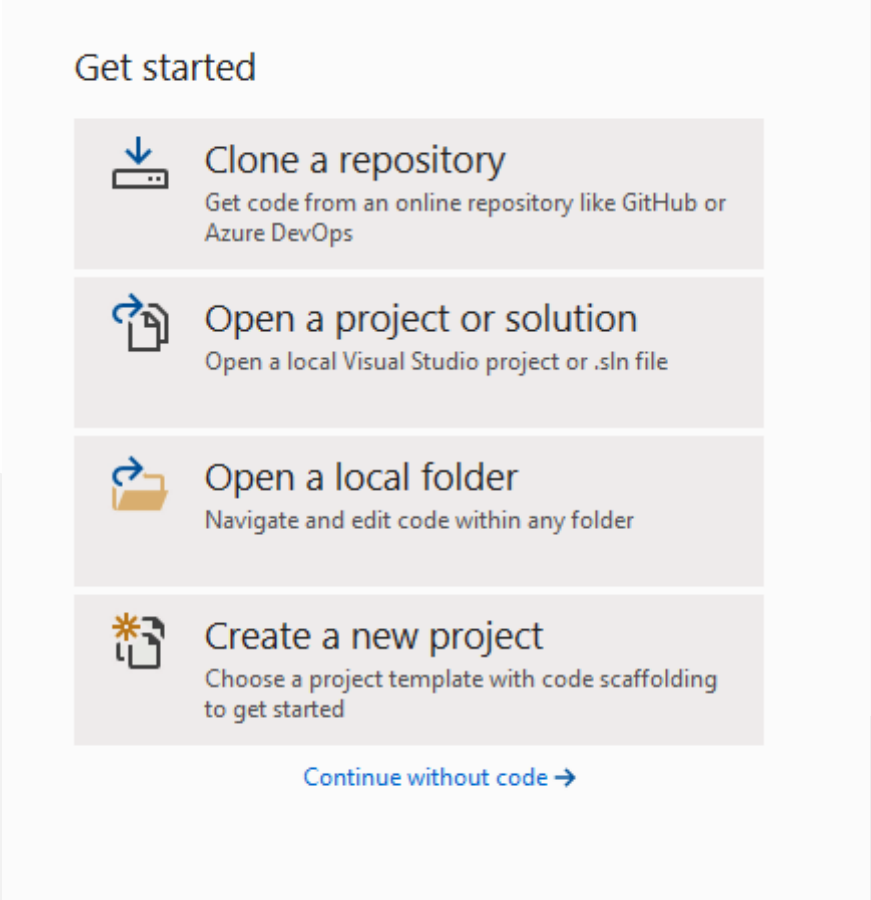
- Open visual studio community edition and select create a new project





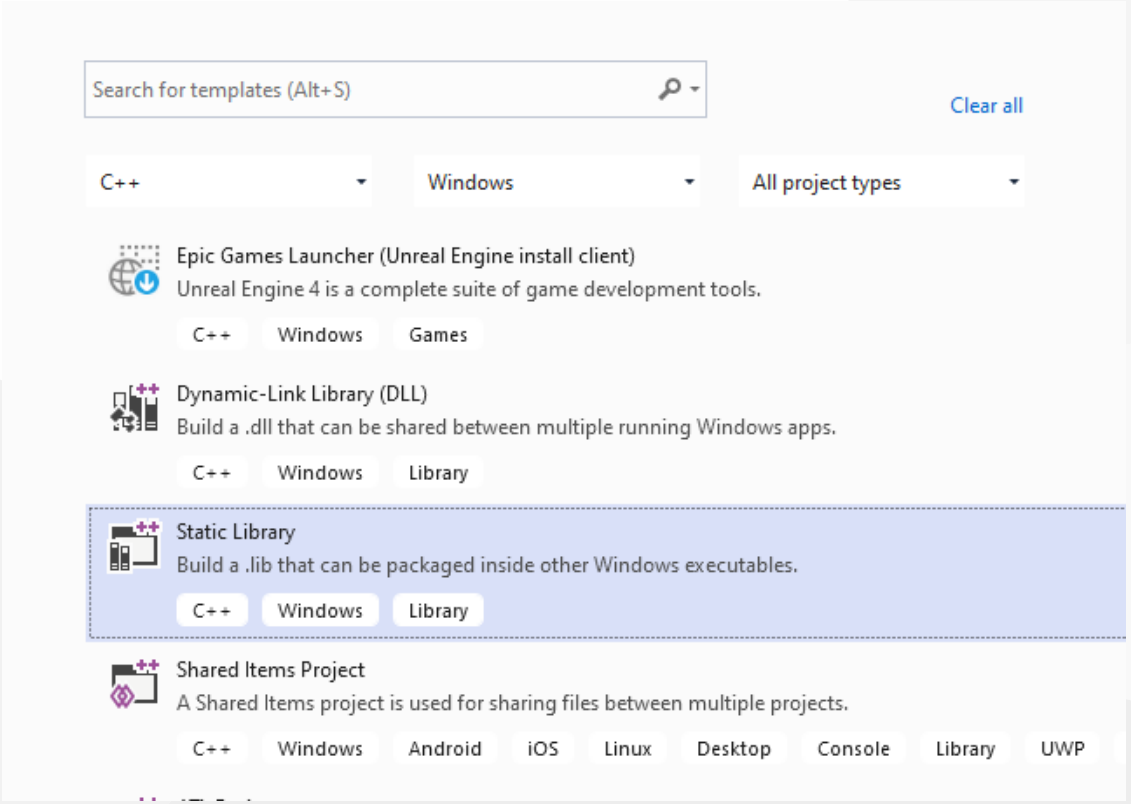
Shared Library Development - (VS C Static Library)-3

- Select create a new project



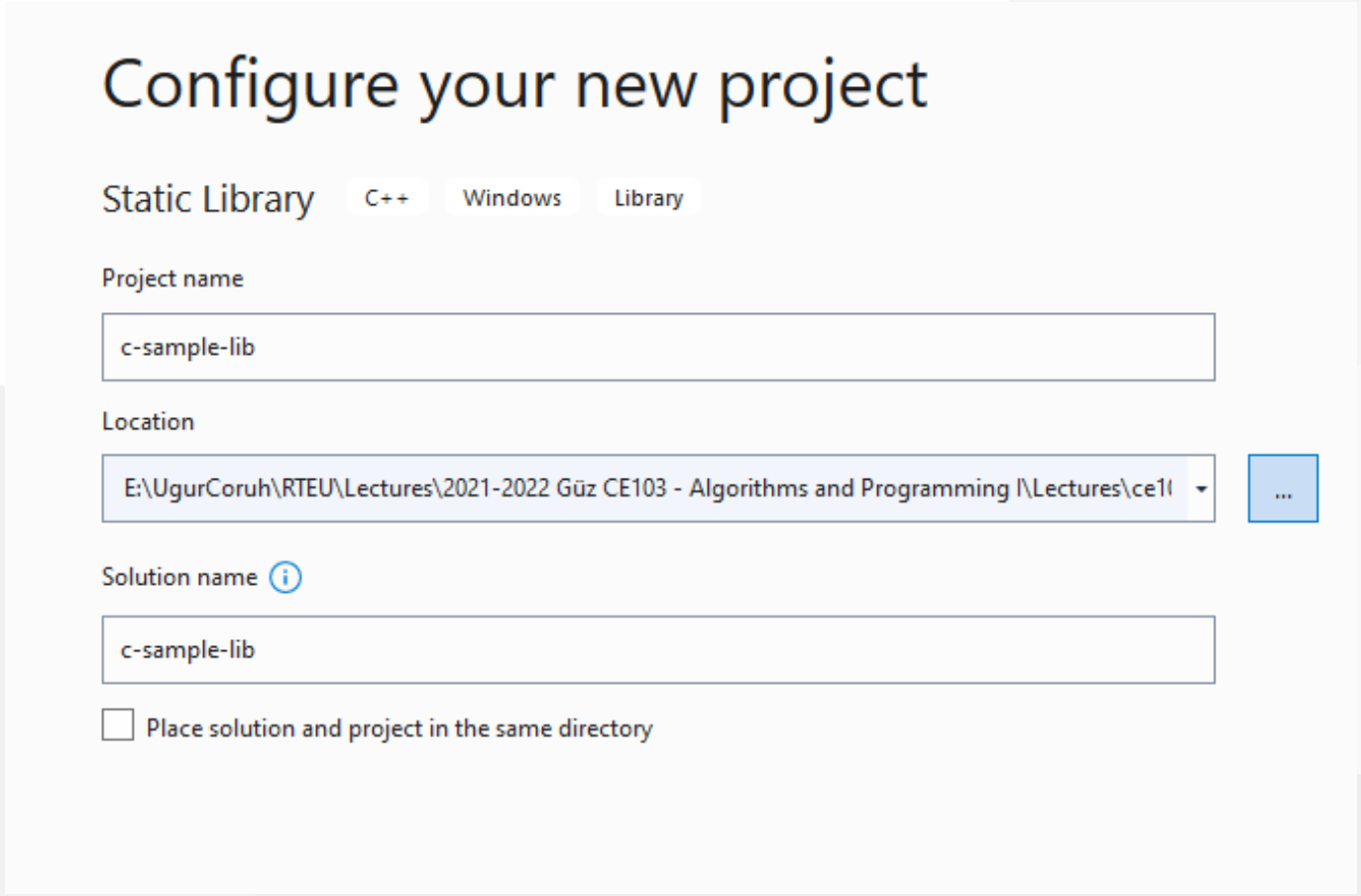
Shared Library Development - (VS C Static Library)-4

- Select C++ static library from the project list



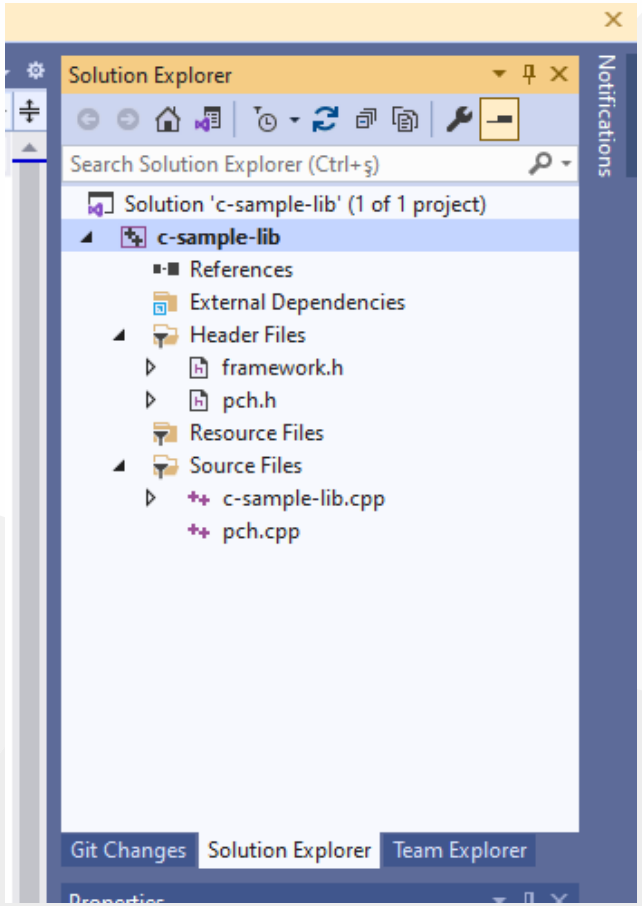
Shared Library Development - (VS C Static Library)-5

- Give static library project name



Shared Library Development - (VS C Static Library)-6

- Default configuration come with C++ project types and setting



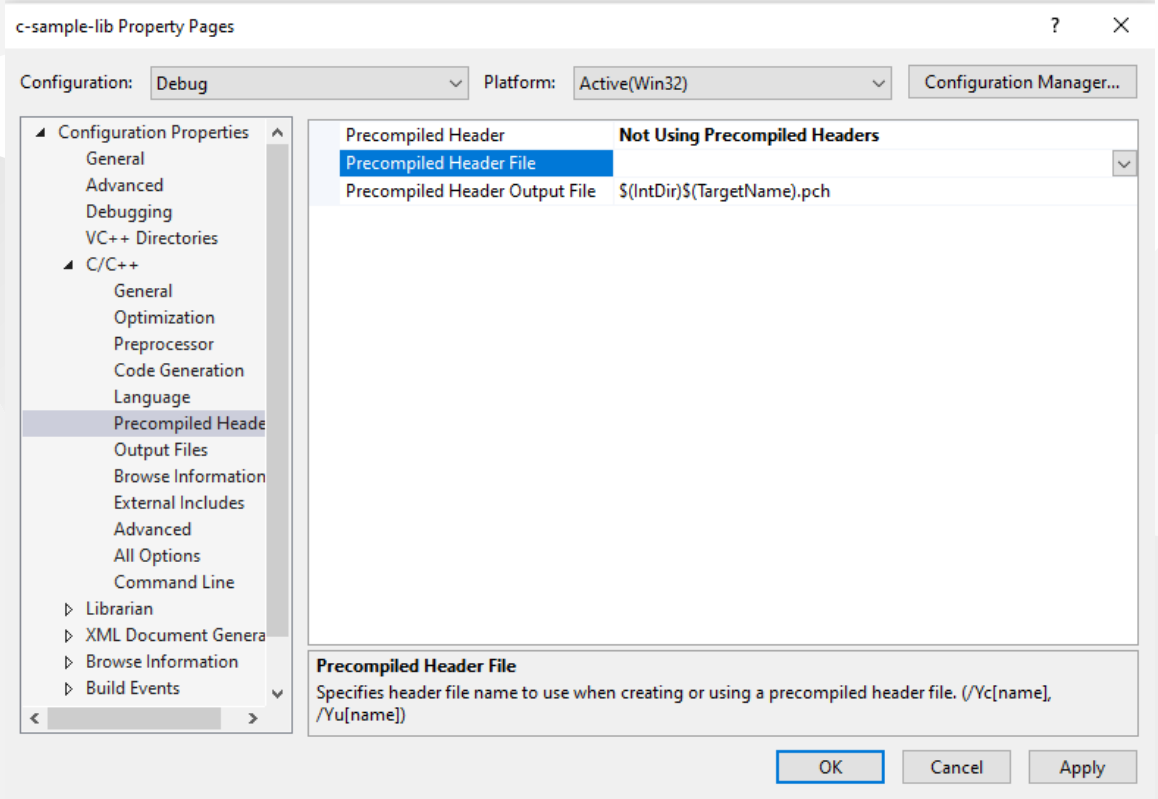
Shared Library Development - (VS C Static Library)-7

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

```
void fncsamplelib(){

}
```

Shared Library Development - (VS C Static Library)-8

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.



Shared Library Development - (VS C Static Library)-9

- 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;
}
```

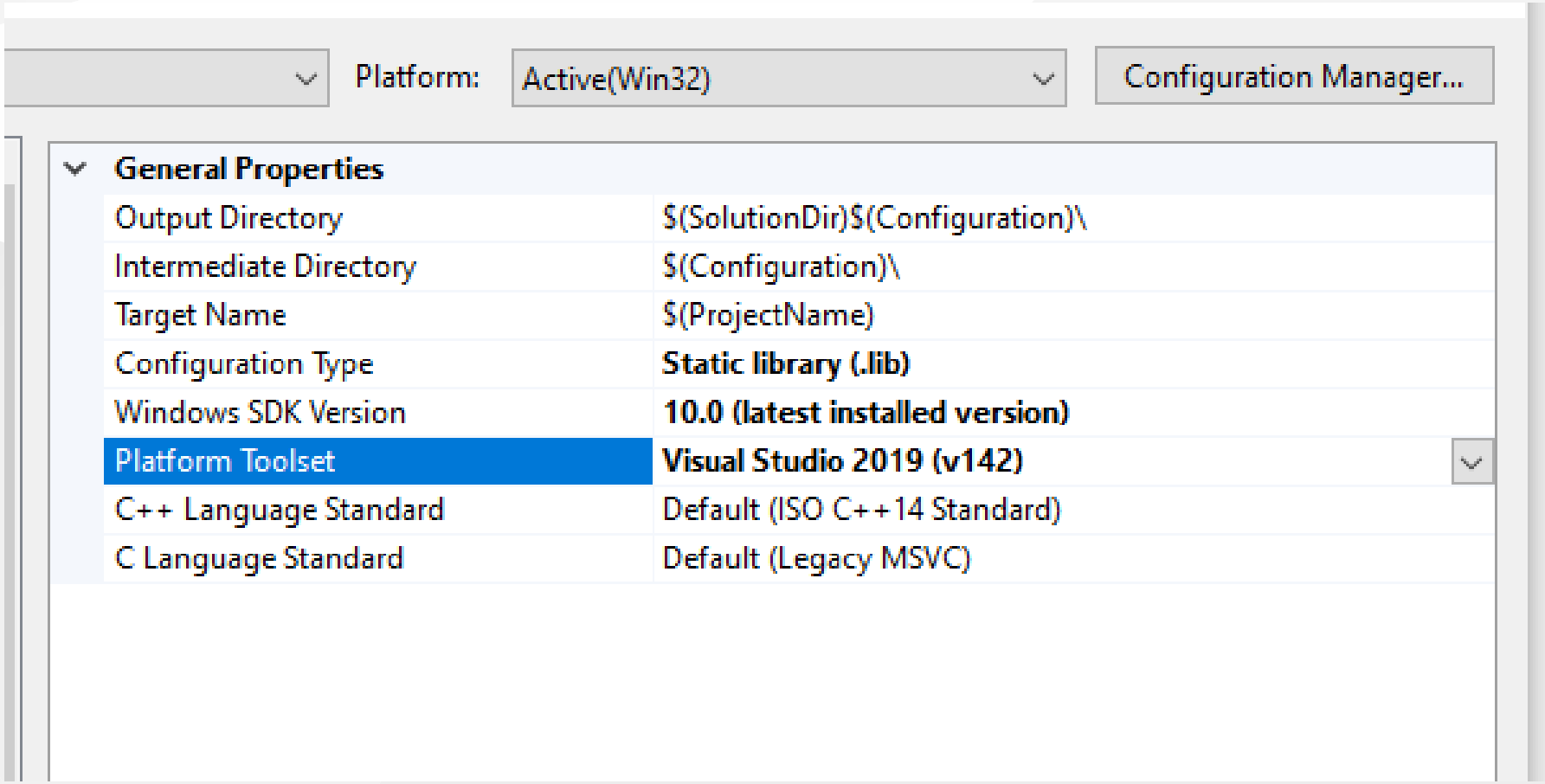
Shared Library Development - (VS C Static Library)-10

- Also, update `samplelib.h` as follows.

```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);
```

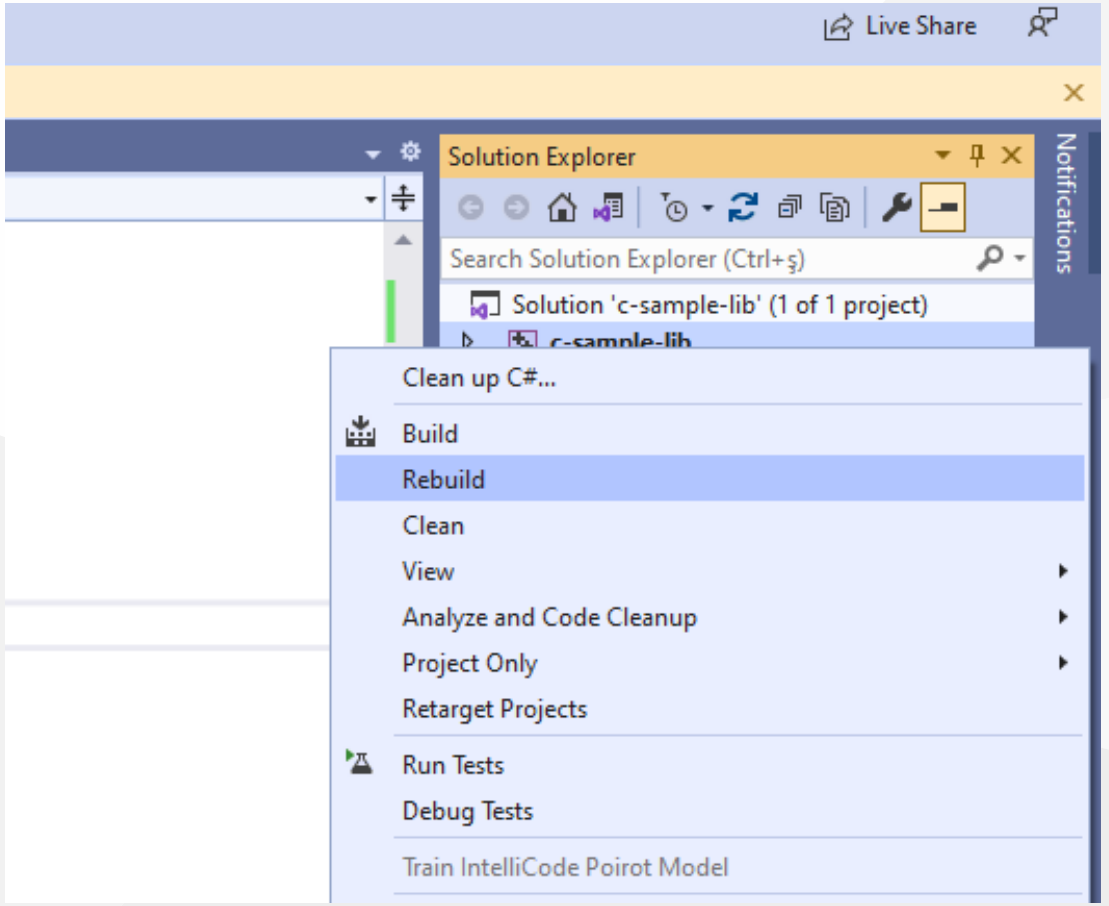
Shared Library Development - (VS C Static Library)-11

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



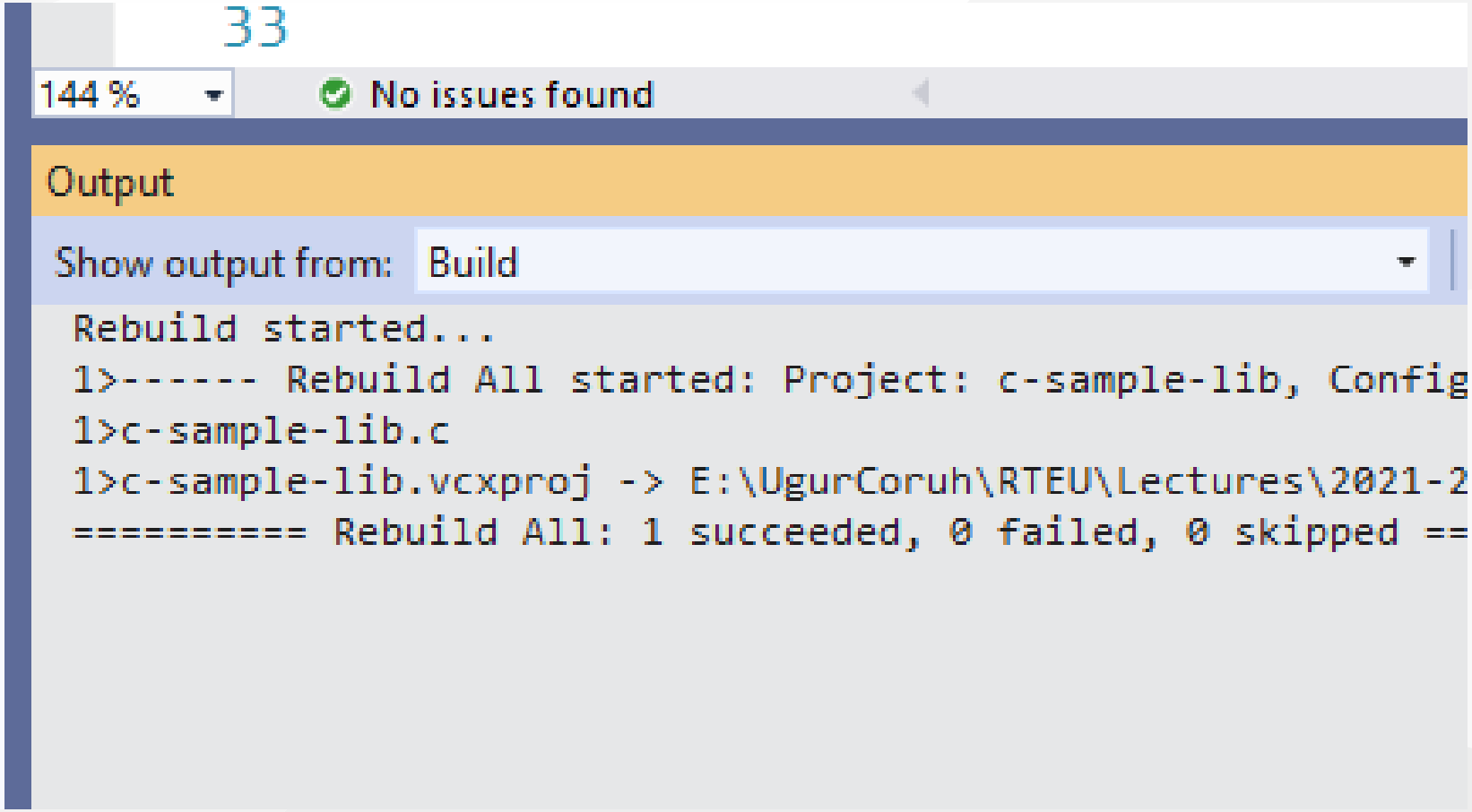
Shared Library Development - (VS C Static Library)-12

- Now we can compile our library



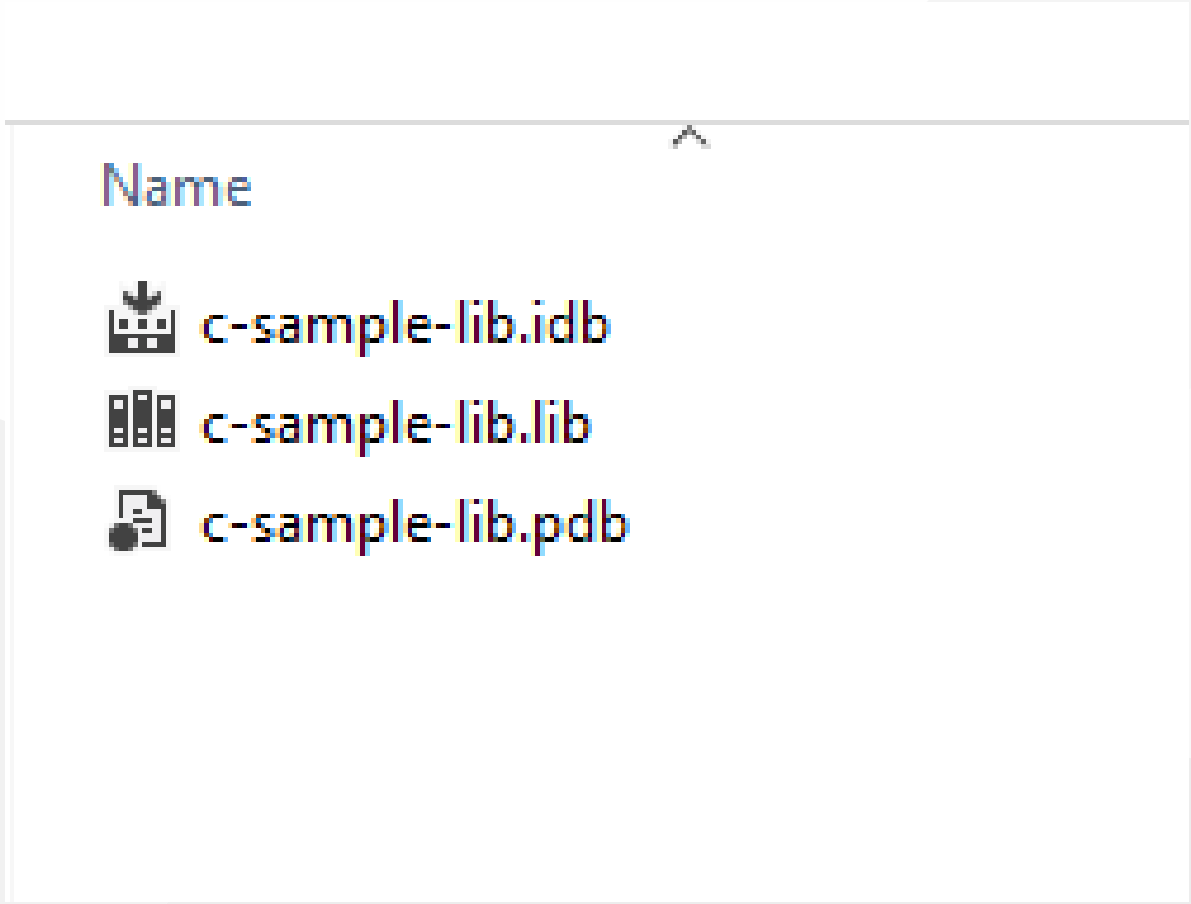
Shared Library Development - (VS C Static Library)-13

- You can follow operation from the output window



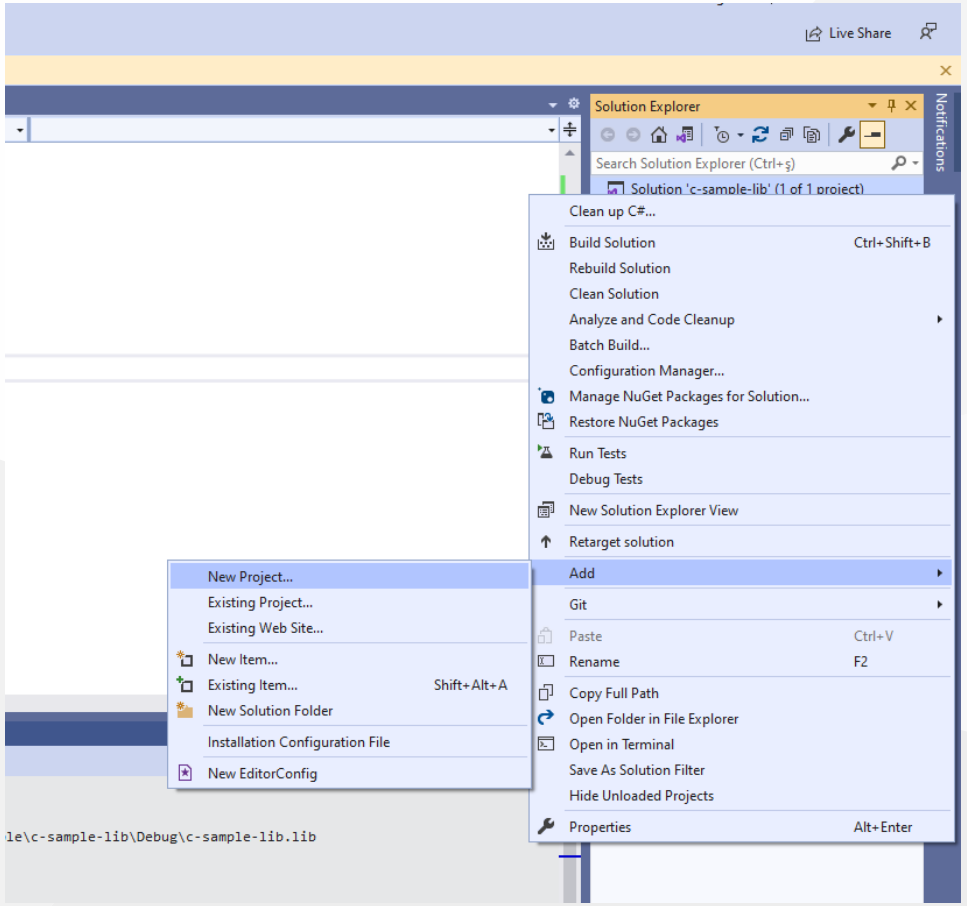
Shared Library Development - (VS C Static Library)-14

- In the debug folder, we will see our output



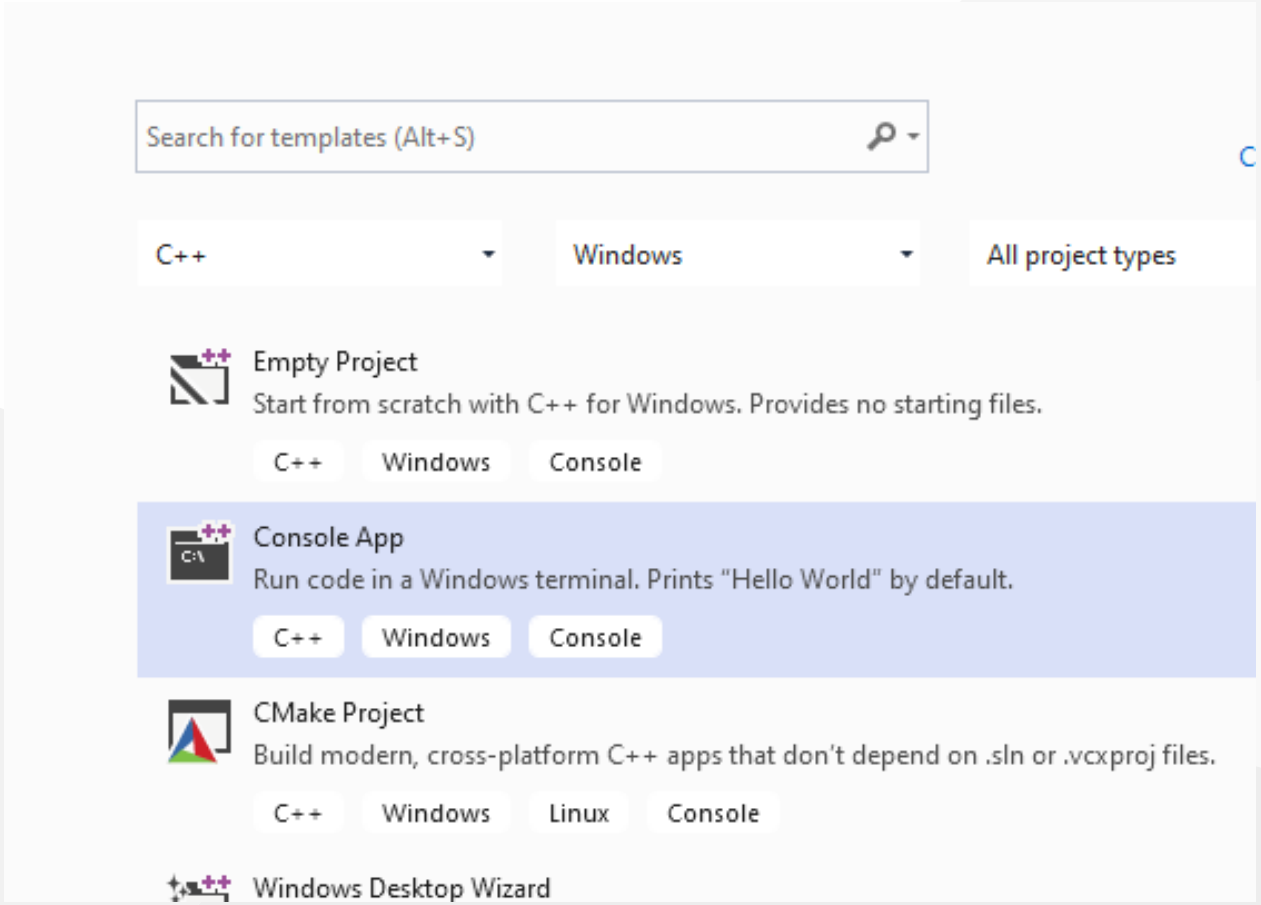
Shared Library Development - (VS C Static Library)-15

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



Shared Library Development - (VS C Static Library)-16

select C++ Windows Console Application from list



Shared Library Development - (VS C Static Library)-17

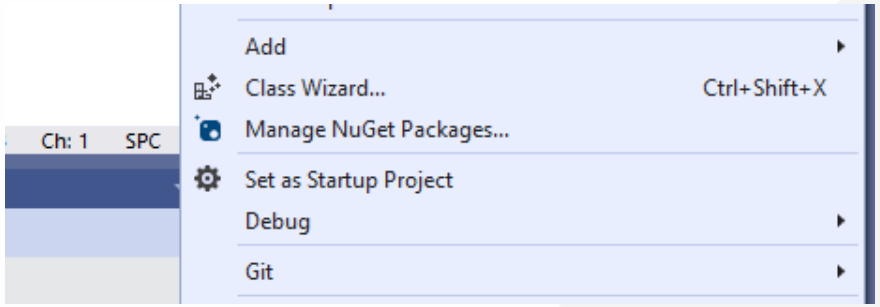
- C++ Console Application Selection will generate a C++ console project we can change extension to C to compile our application as C application.

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

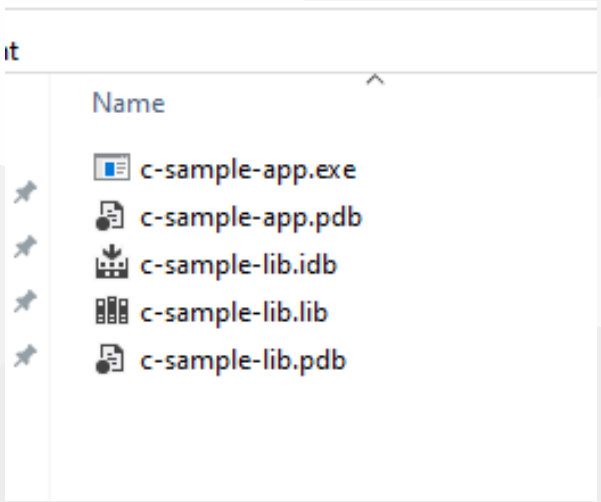
```c
#include <stdio.h>

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

Shared Library Development - (VS C Static Library)-18

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 the application we will see only `"Hello World"`

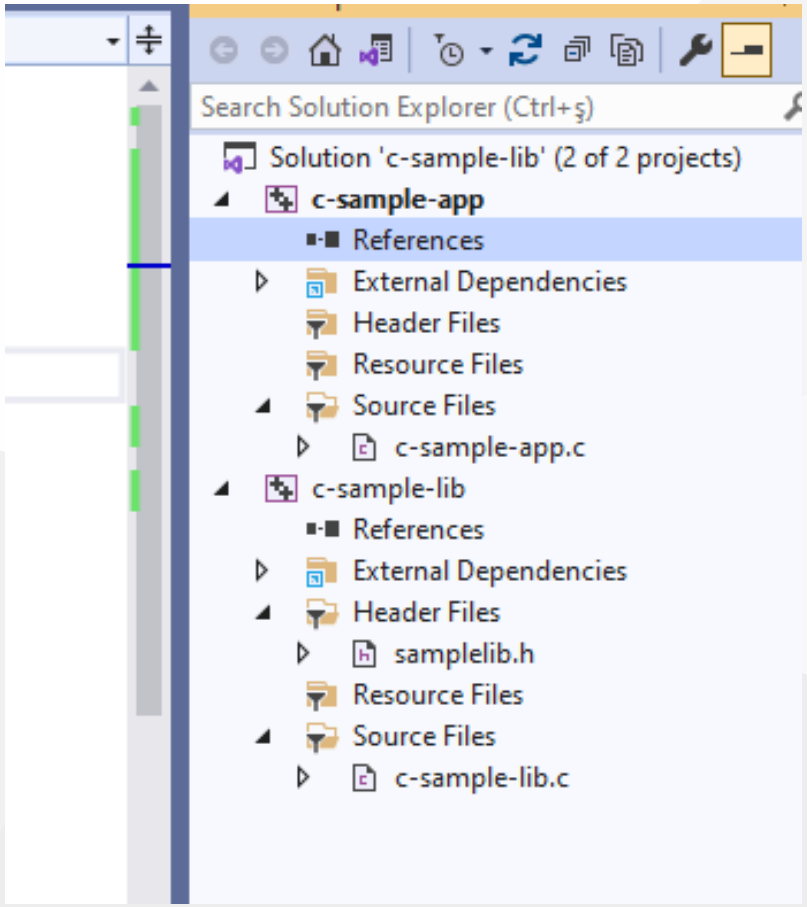
Shared Library Development - (VS C Static Library)-19

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

Shared Library Development - (VS C Static Library)-20

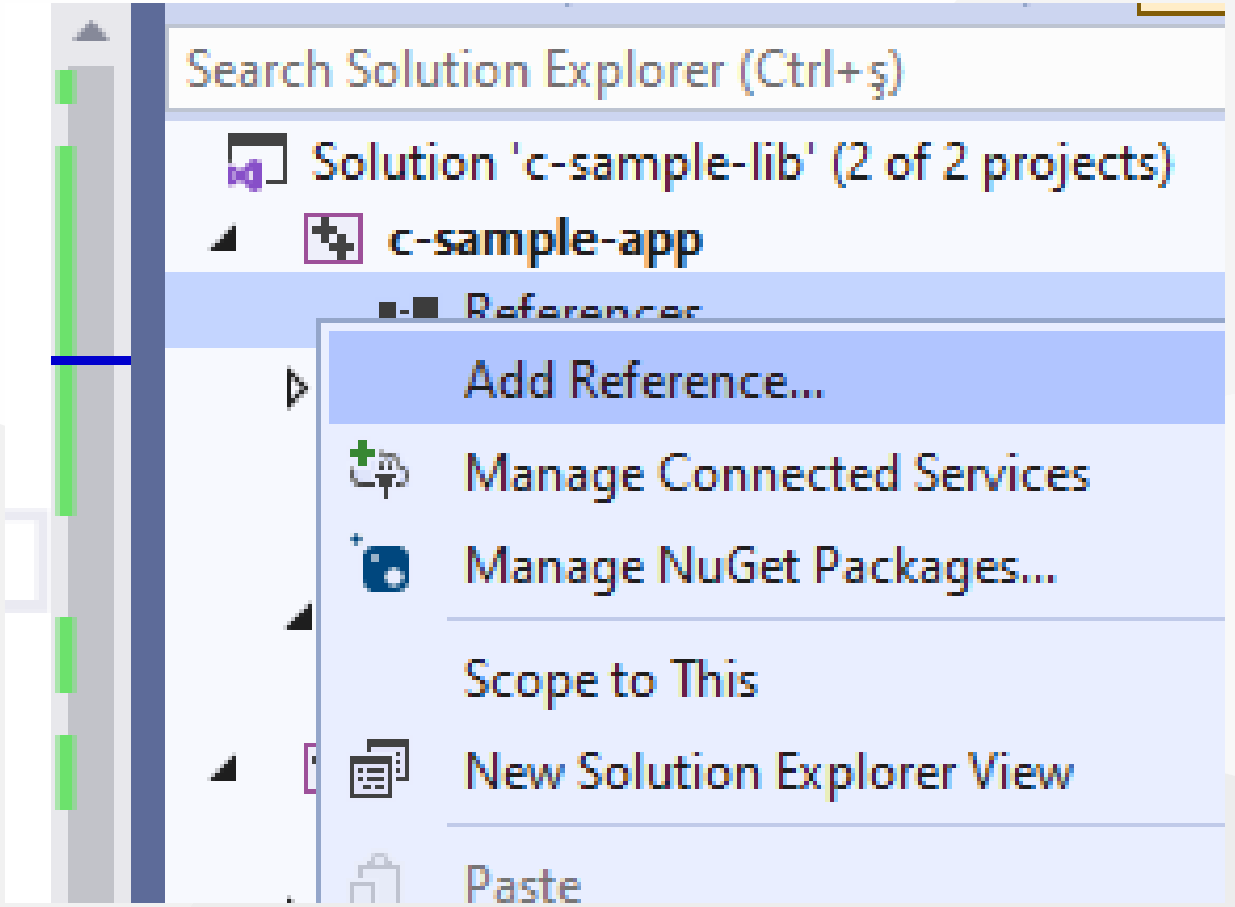
First option

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



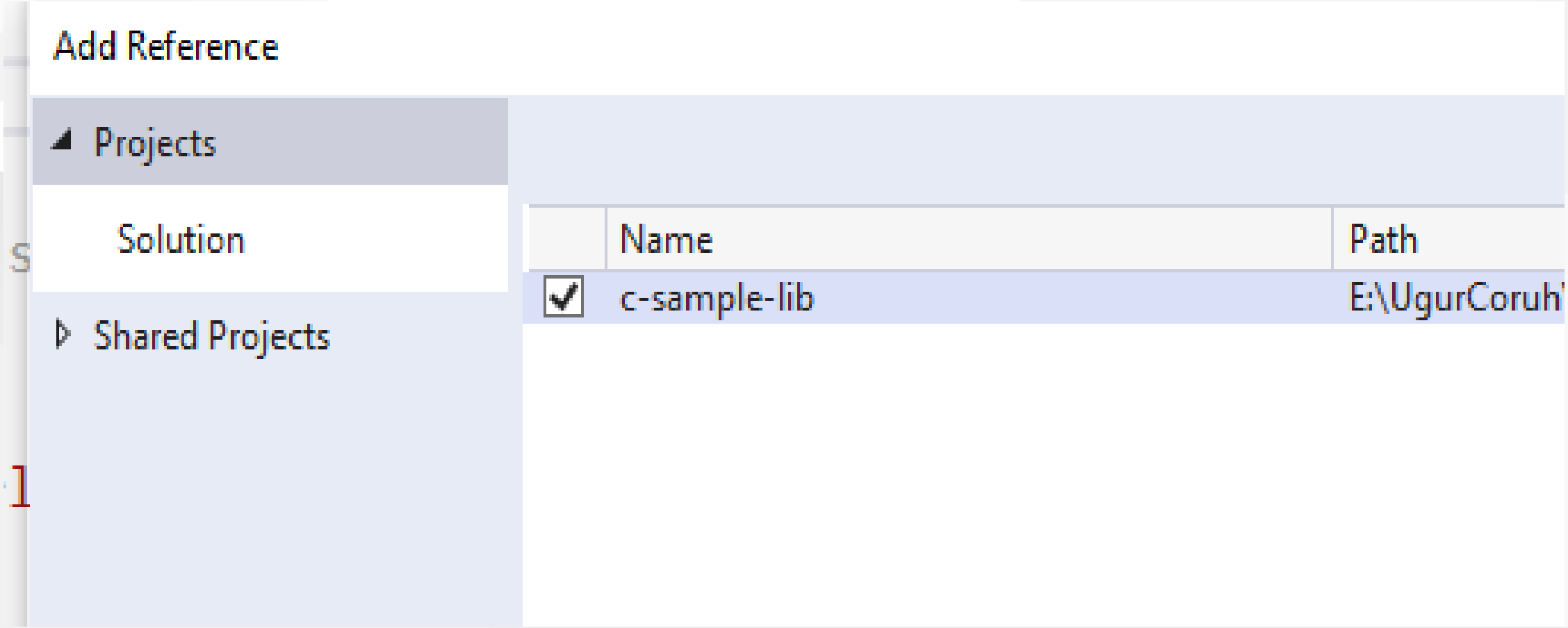
Shared Library Development - (VS C Static Library)-21

- Select Add Reference



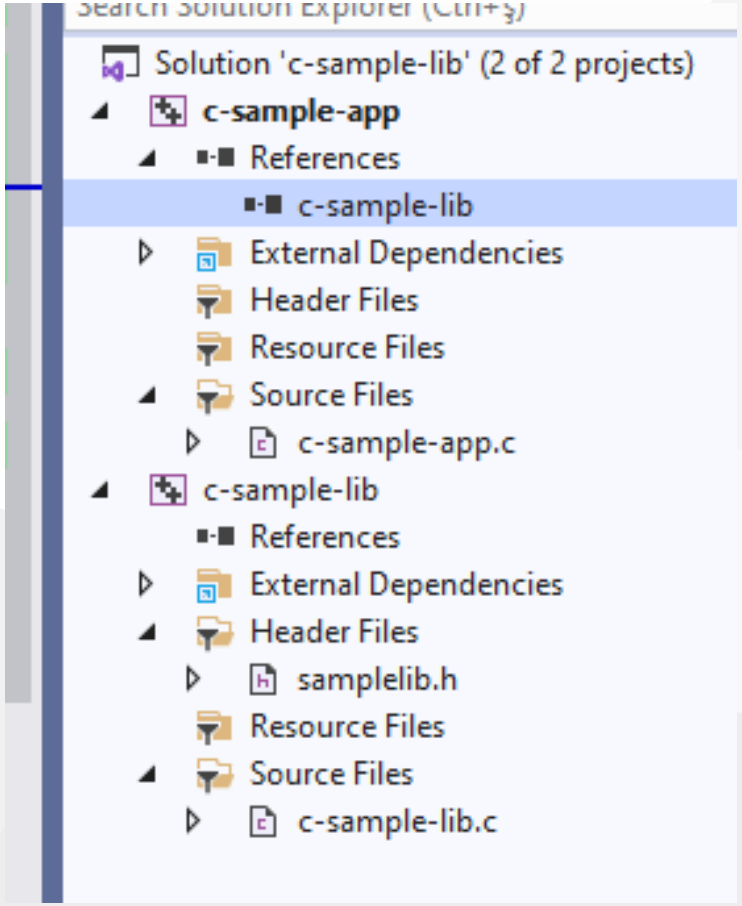
Shared Library Development - (VS C Static Library)-22

- Browse for solution and select `c-sample-lib`



Shared Library Development - (VS C Static Library)-23

You can check added reference from references section



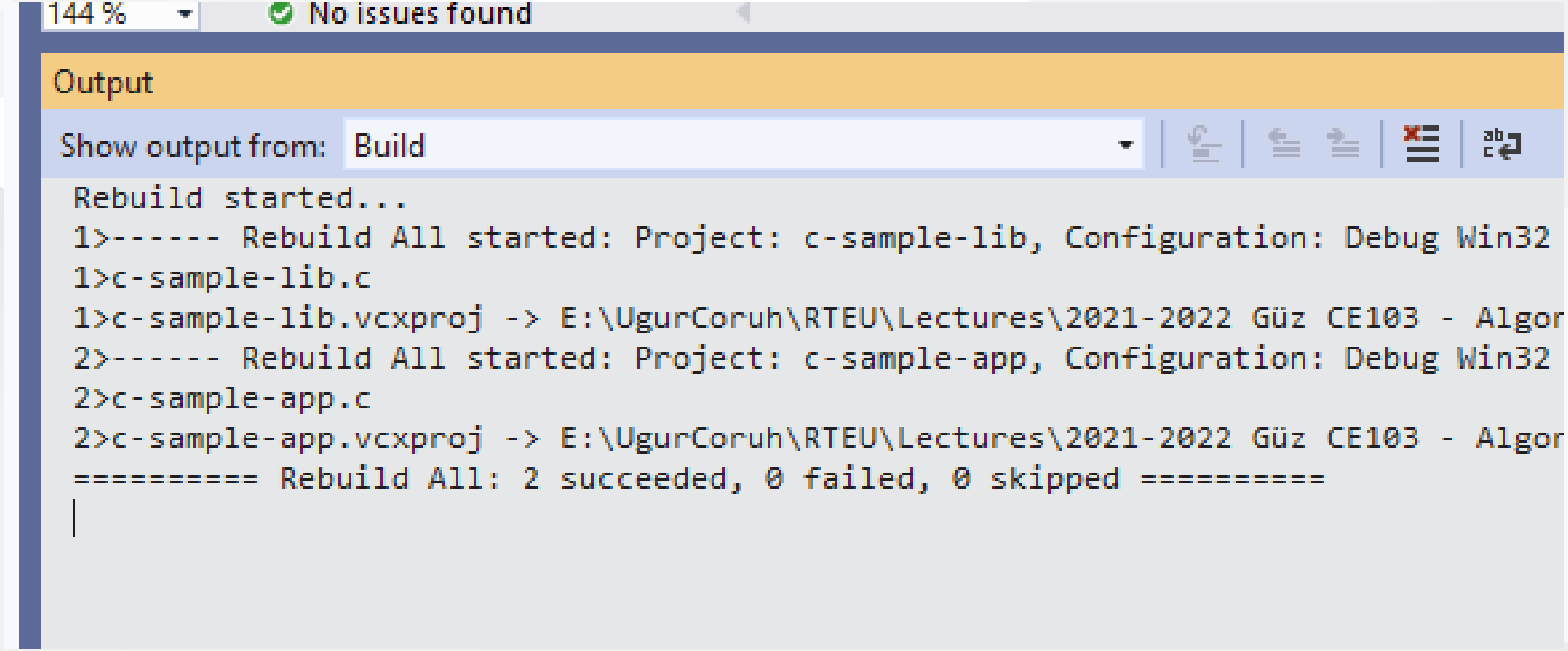
Shared Library Development - (VS C Static Library)-24

- 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

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

Shared Library Development - (VS C Static Library)-25

- we can build our `c-sample-app`



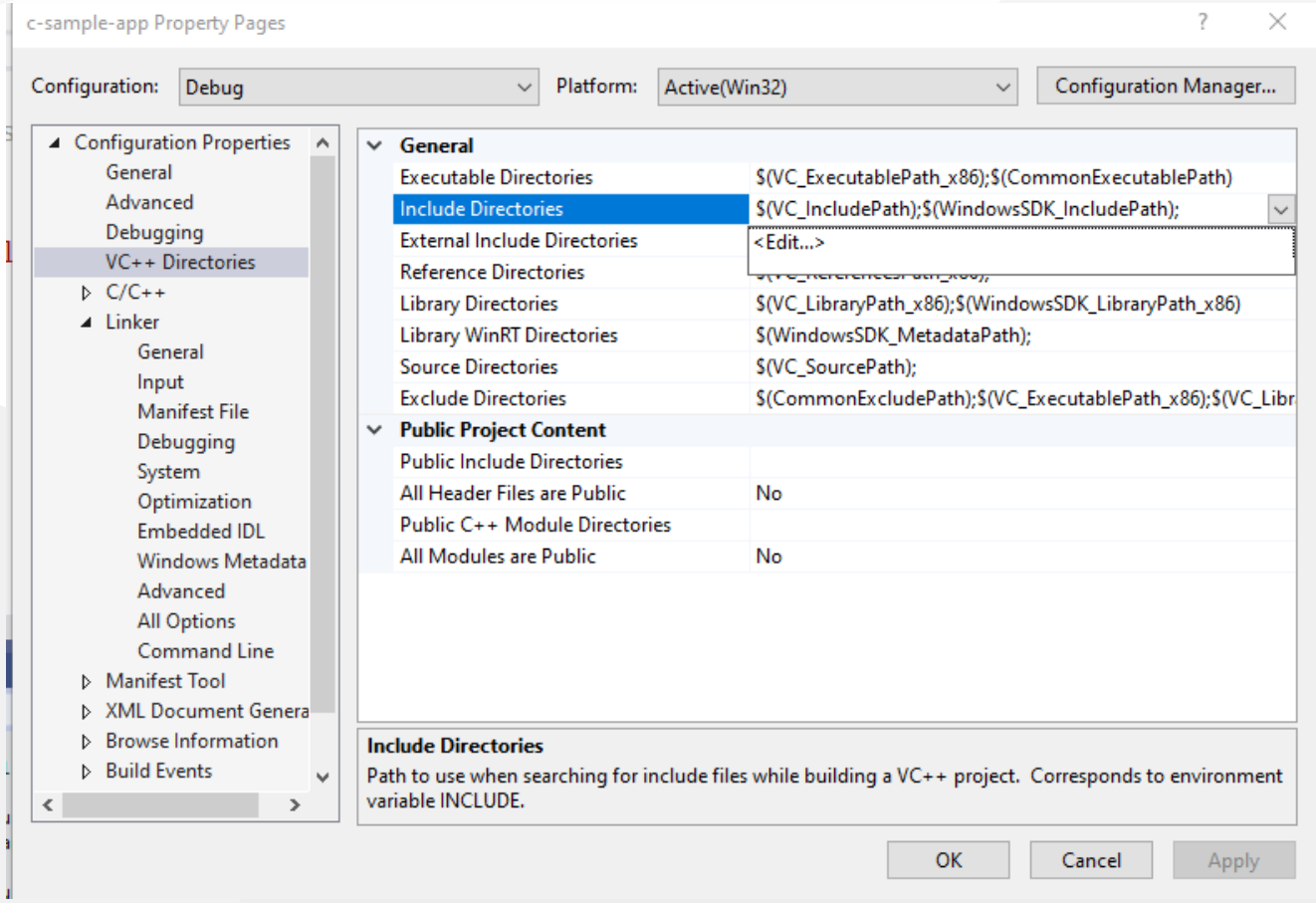
Shared Library Development - (VS C Static Library)-26

- Also we can only write header name

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

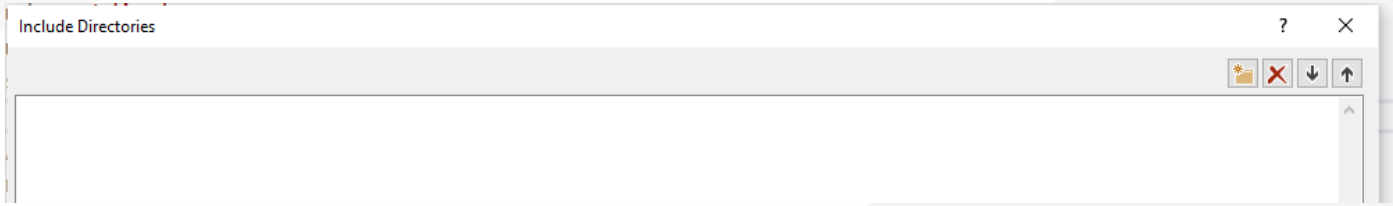
Shared Library Development - (VS C Static Library)-27

- For this option, we need to configure include directories

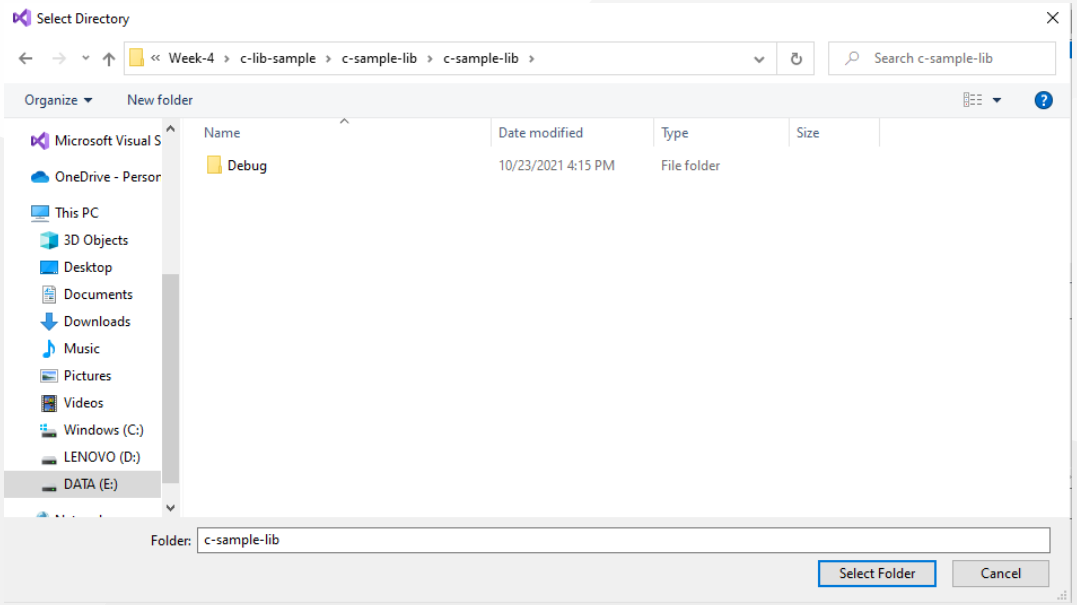


Shared Library Development - (VS C Static Library)-28

select c-sample-lib header file location

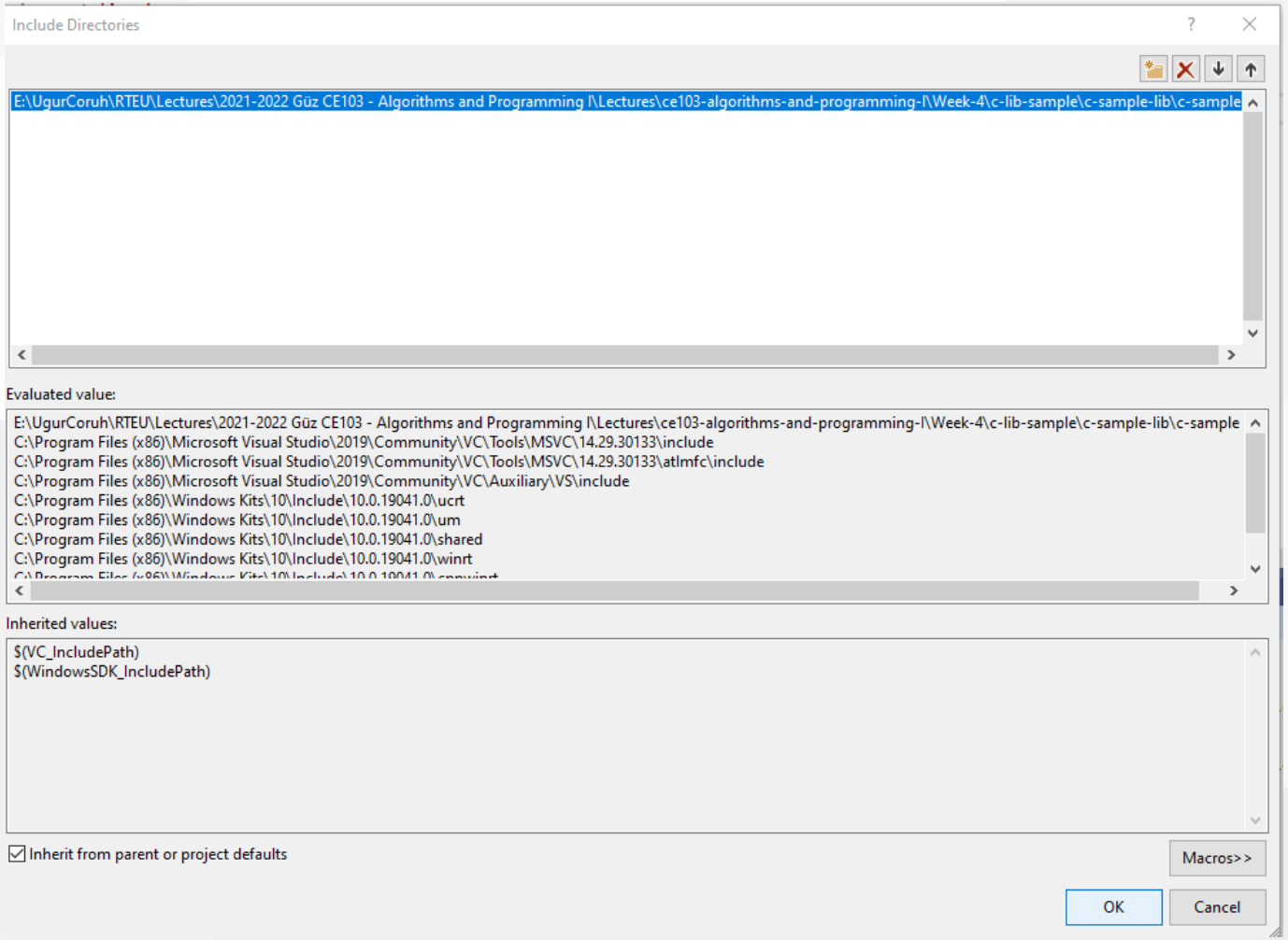


browse for folder



Shared Library Development - (VS C Static Library)-29

your full path will be added to your configuration



Shared Library Development - (VS C Static Library)-30

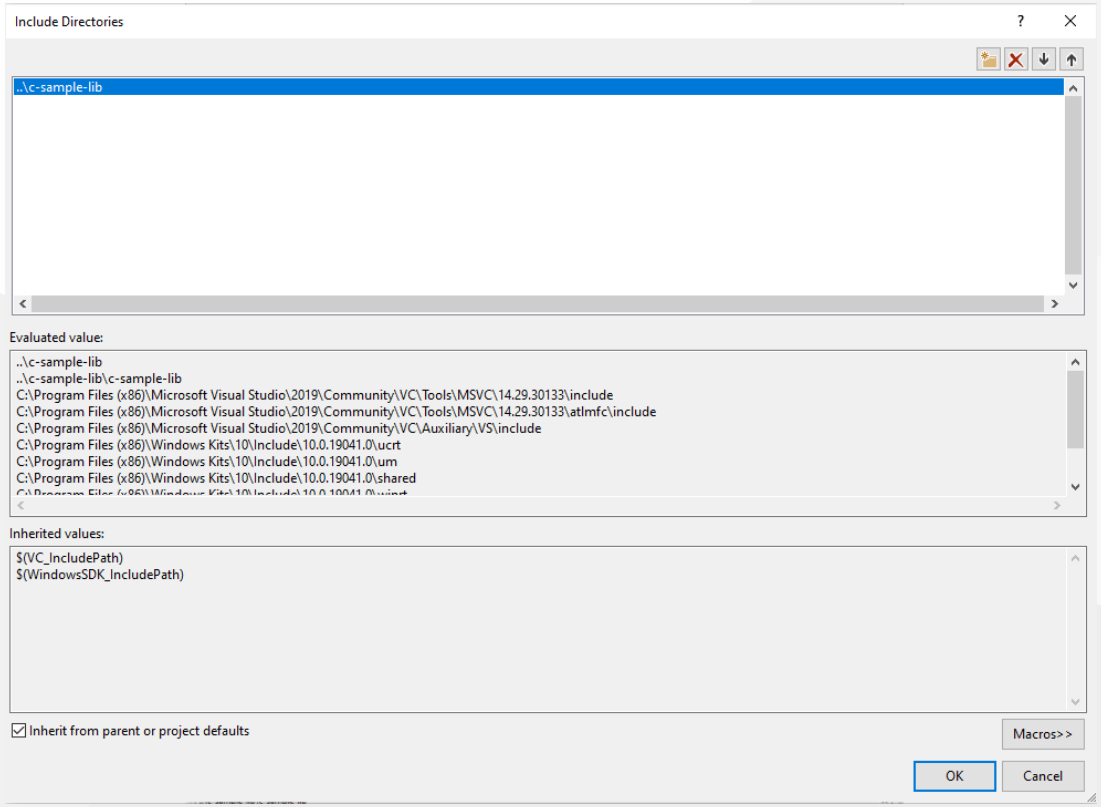
if you add header file paths to your configuration you can use header files by name in your source code

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

Shared Library Development - (VS C Static Library)-31

- 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
```



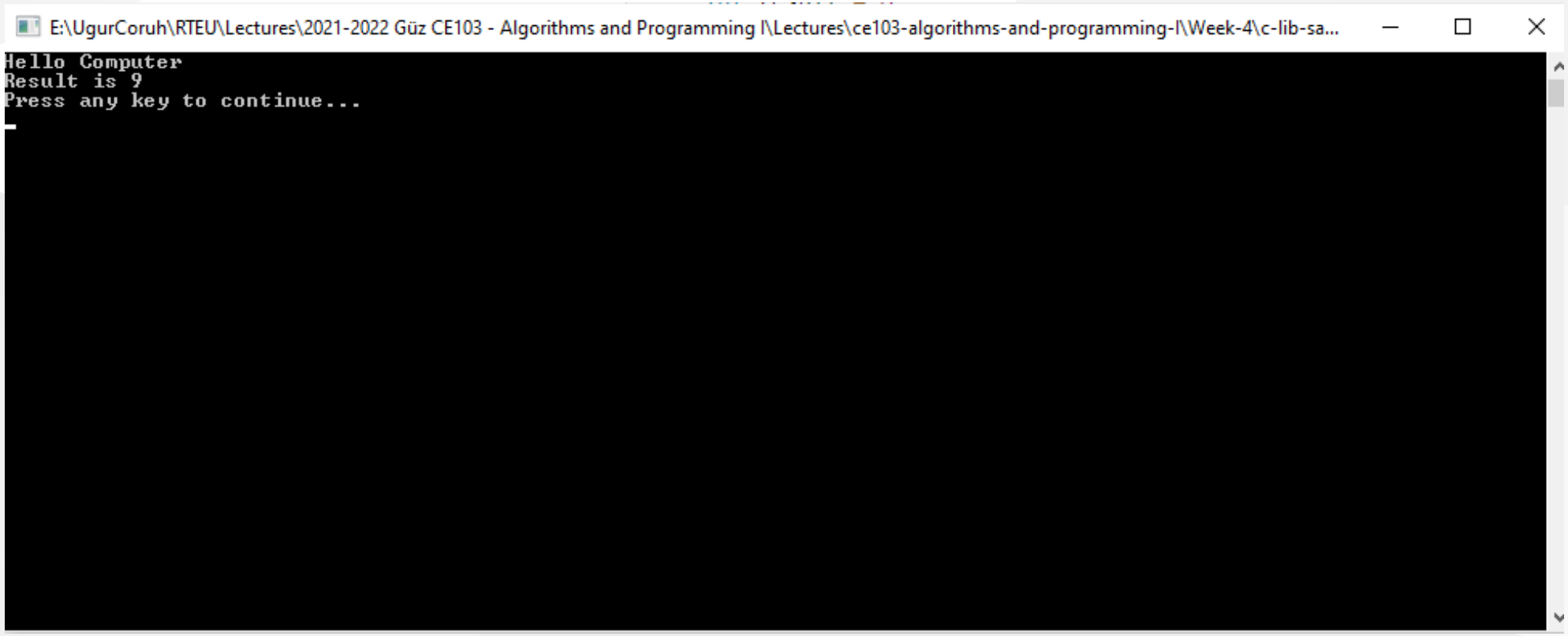
Shared Library Development - (VS C Static Library)-32

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

```c
#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;
}
```

Shared Library Development - (VS C Static Library)-33

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



Shared Library Development - (VS C Static Library)-34

- A 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. In another case you can use a precompiled static library with you or this library can be part of any installation then if there is an installed app and static libraries are placed on the system folder or any different location then you can use configuration files to set library path and included header paths

Shared Library Development - (VS C Static Library)-35

- Now we can remove the 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
```

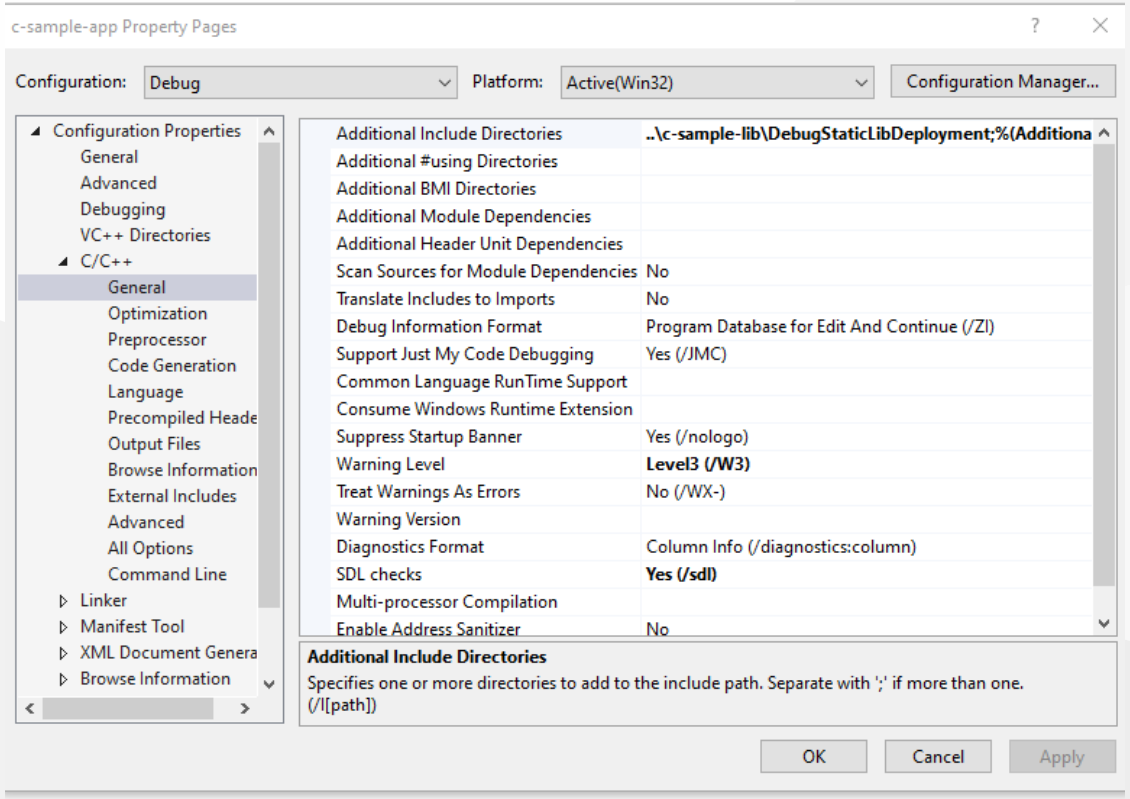
- Set C/C++ -> General -> Additional Include Directories

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

Shared Library Development - (VS C Static Library)-36

Not Working Solution

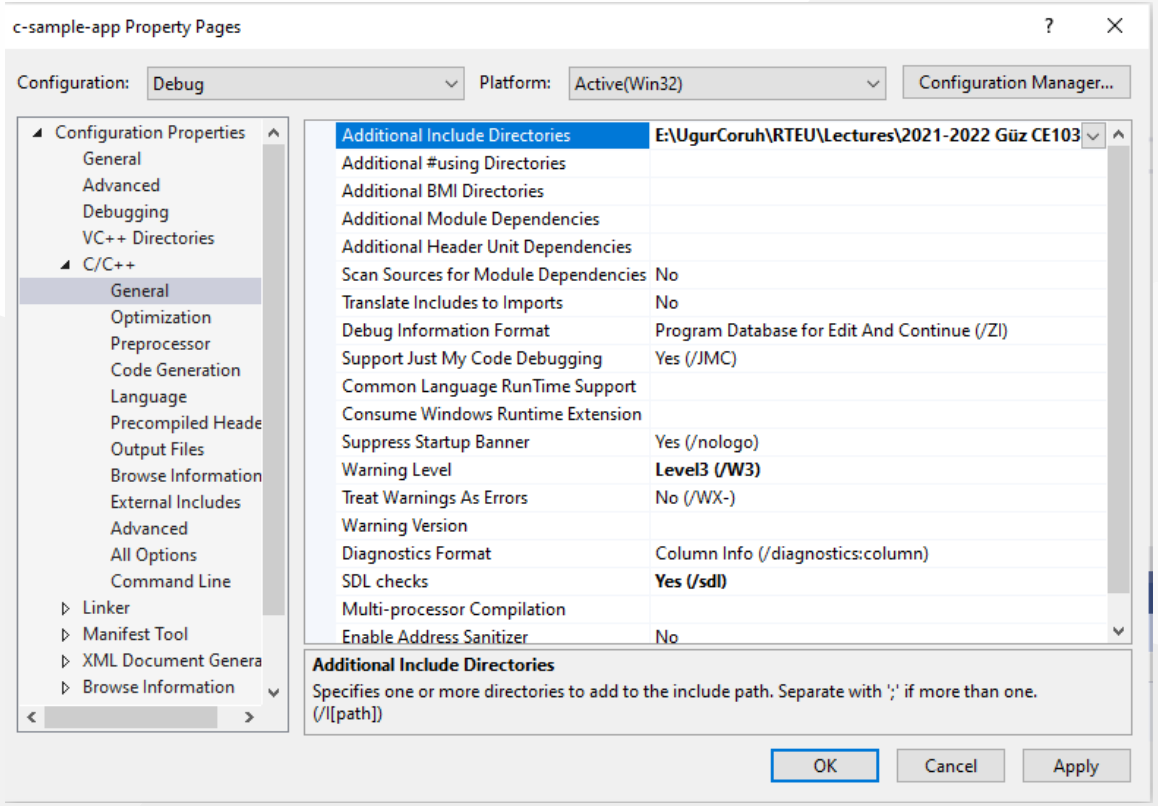
```
..\c-sample-lib\DebugStaticLibDeployment
```



Shared Library Development - (VS C Static Library)-37

Working Solution

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



Shared Library Development - (VS C Static Library)-38

Now we will set library folder that our static library placed

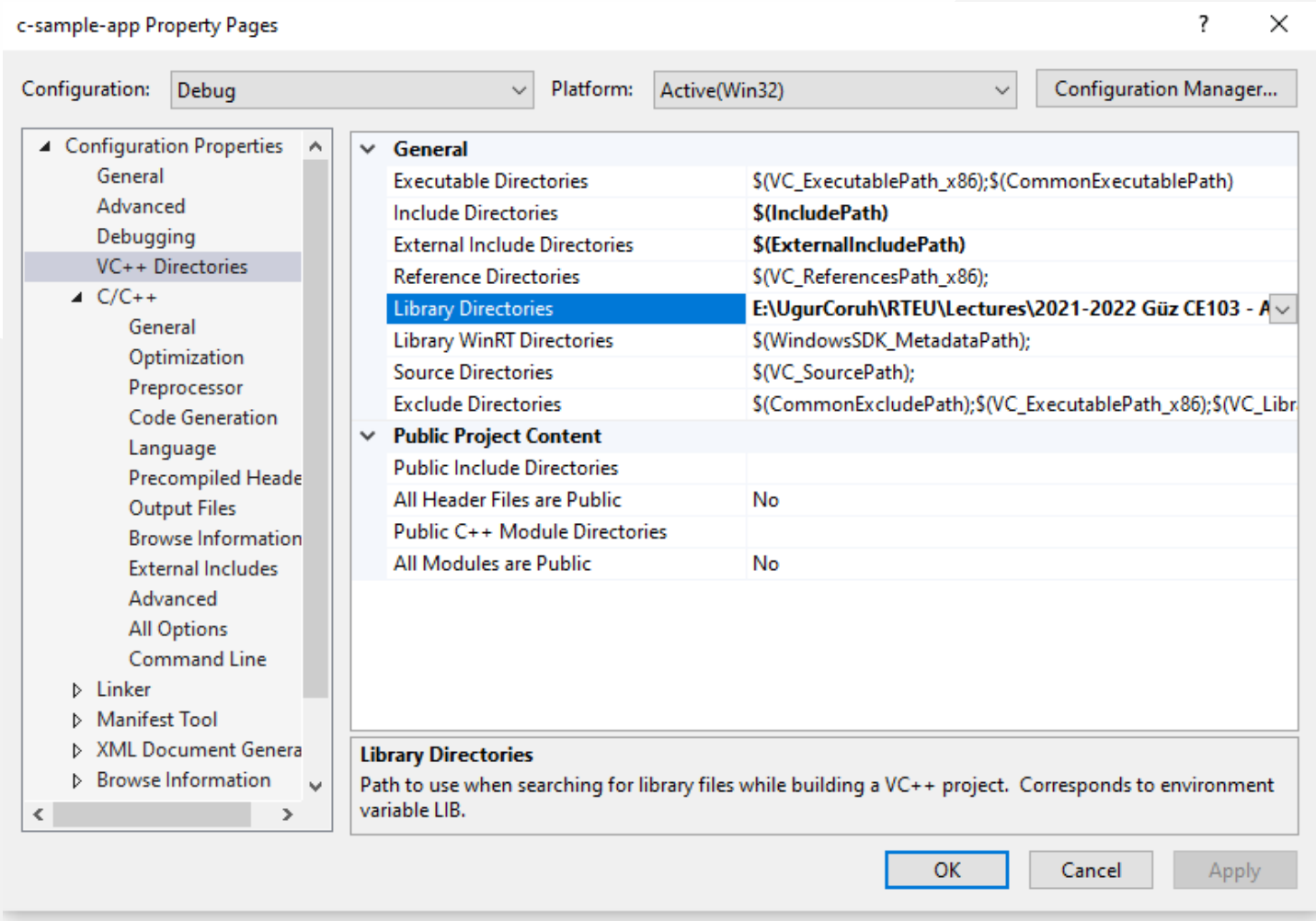
we will set VC++ Directories -> Library Directories

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

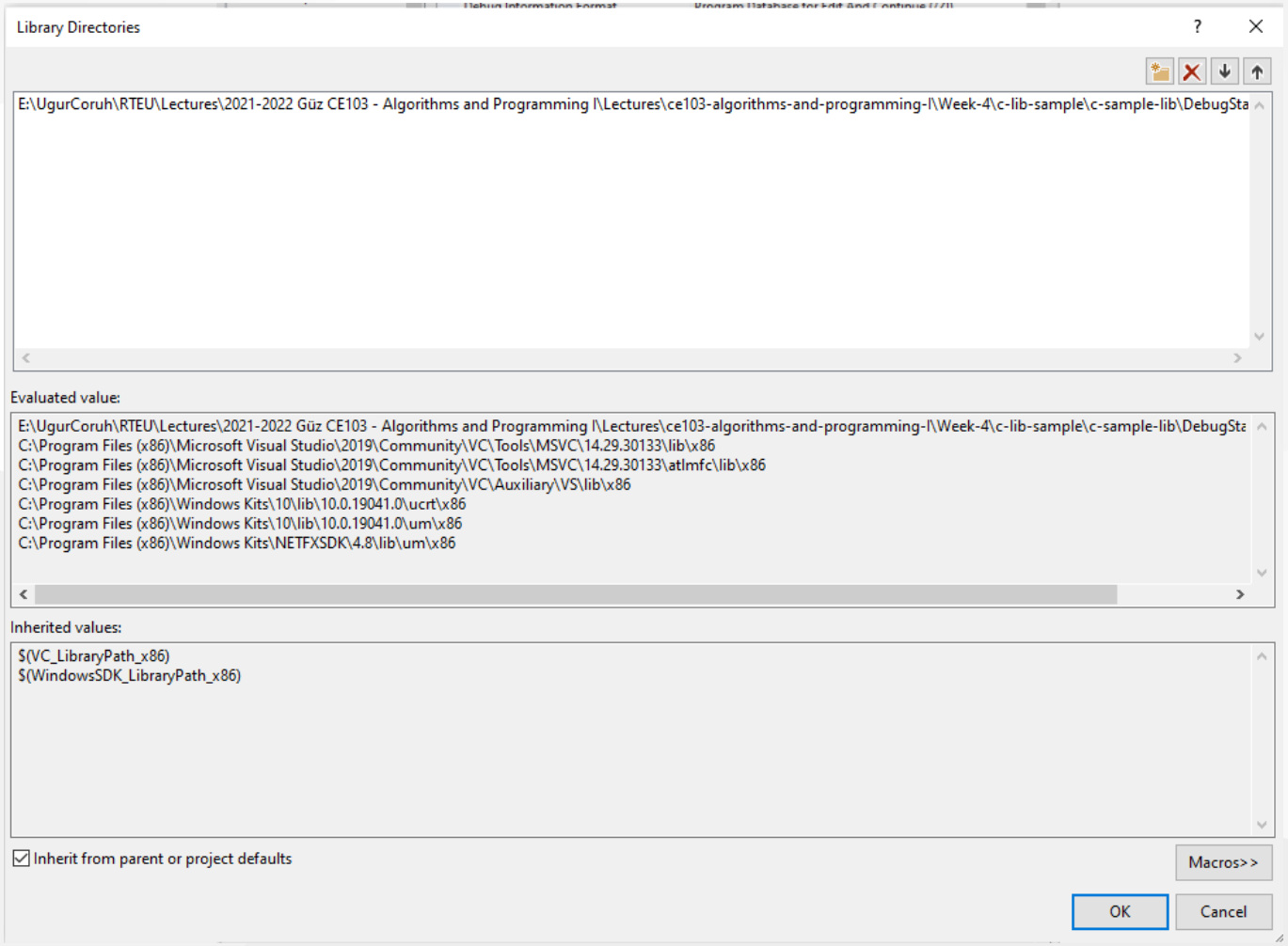
Shared Library Development - (VS C Static Library)-39

Working Solution

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



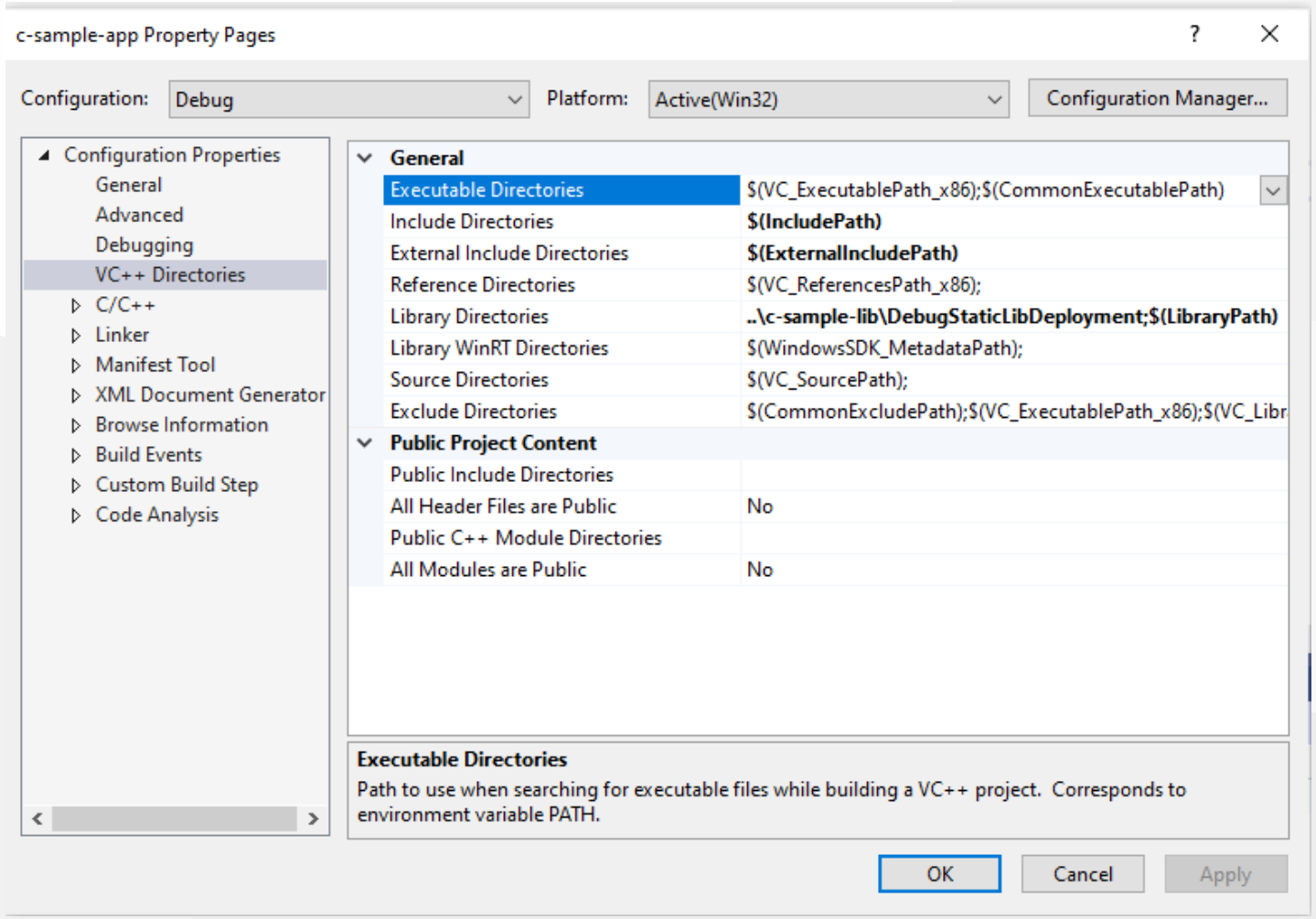
Shared Library Development - (VS C Static Library)-40



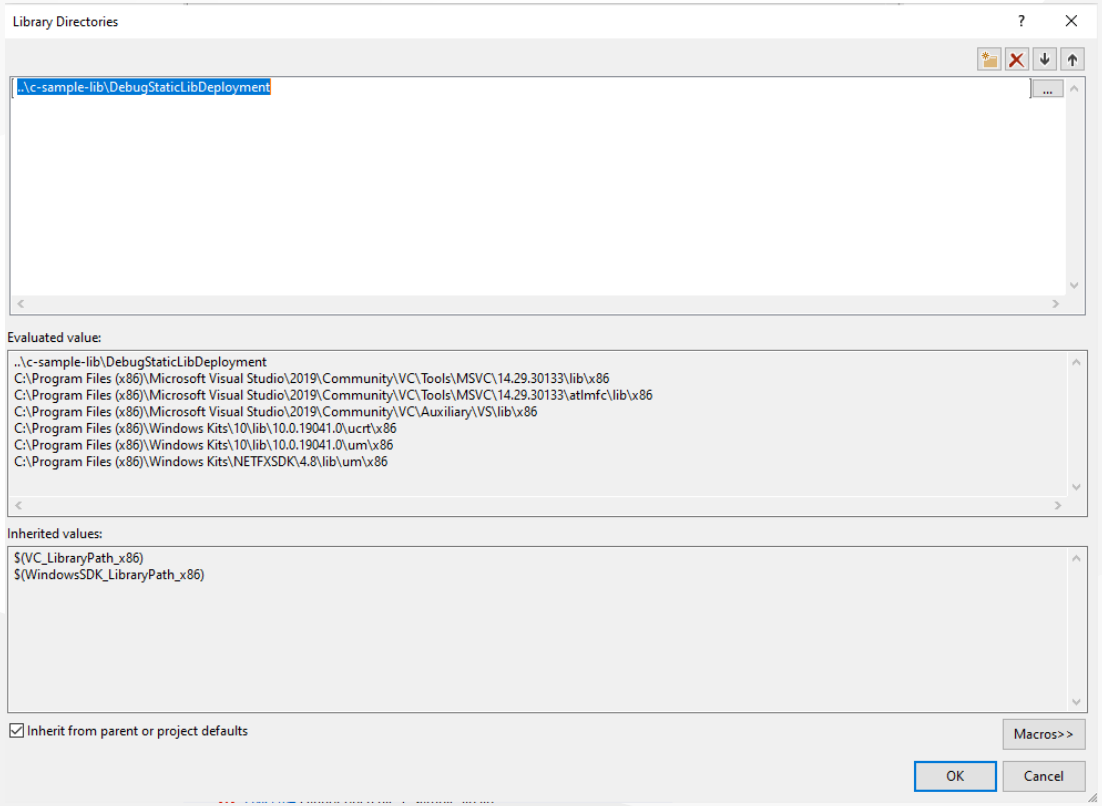
Shared Library Development - (VS C Static Library)-41

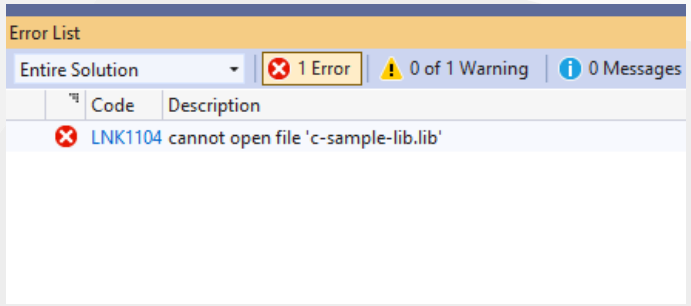
Not Working

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



Shared Library Development - (VS C Static Library)-42

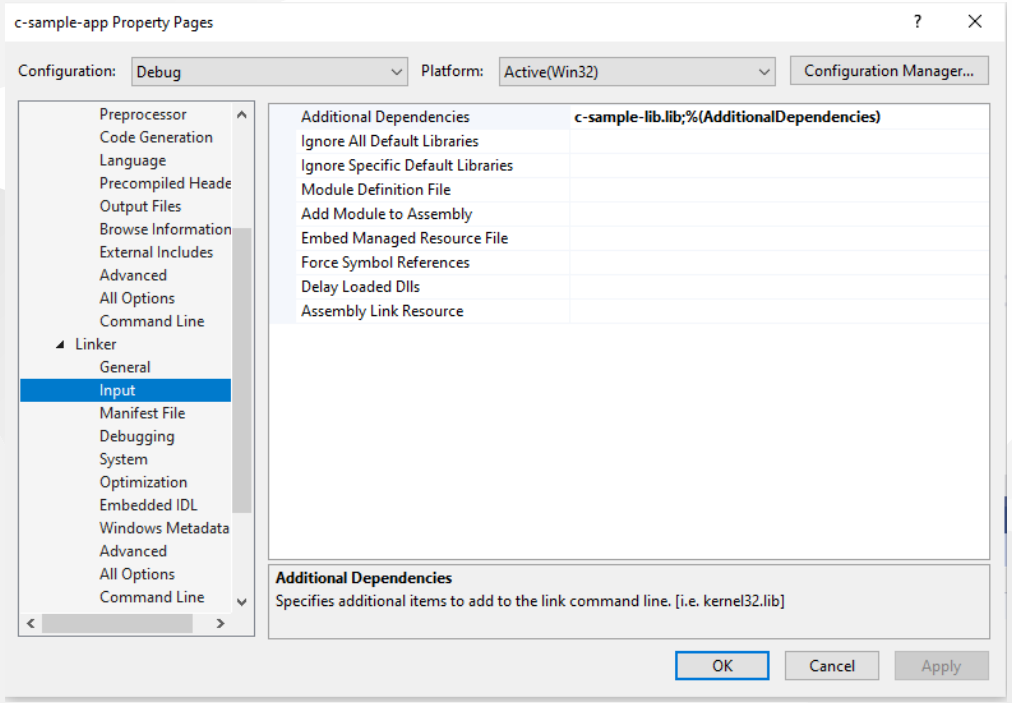




Shared Library Development - (VS C Static Library)-43

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 a different location and they are ready to use.

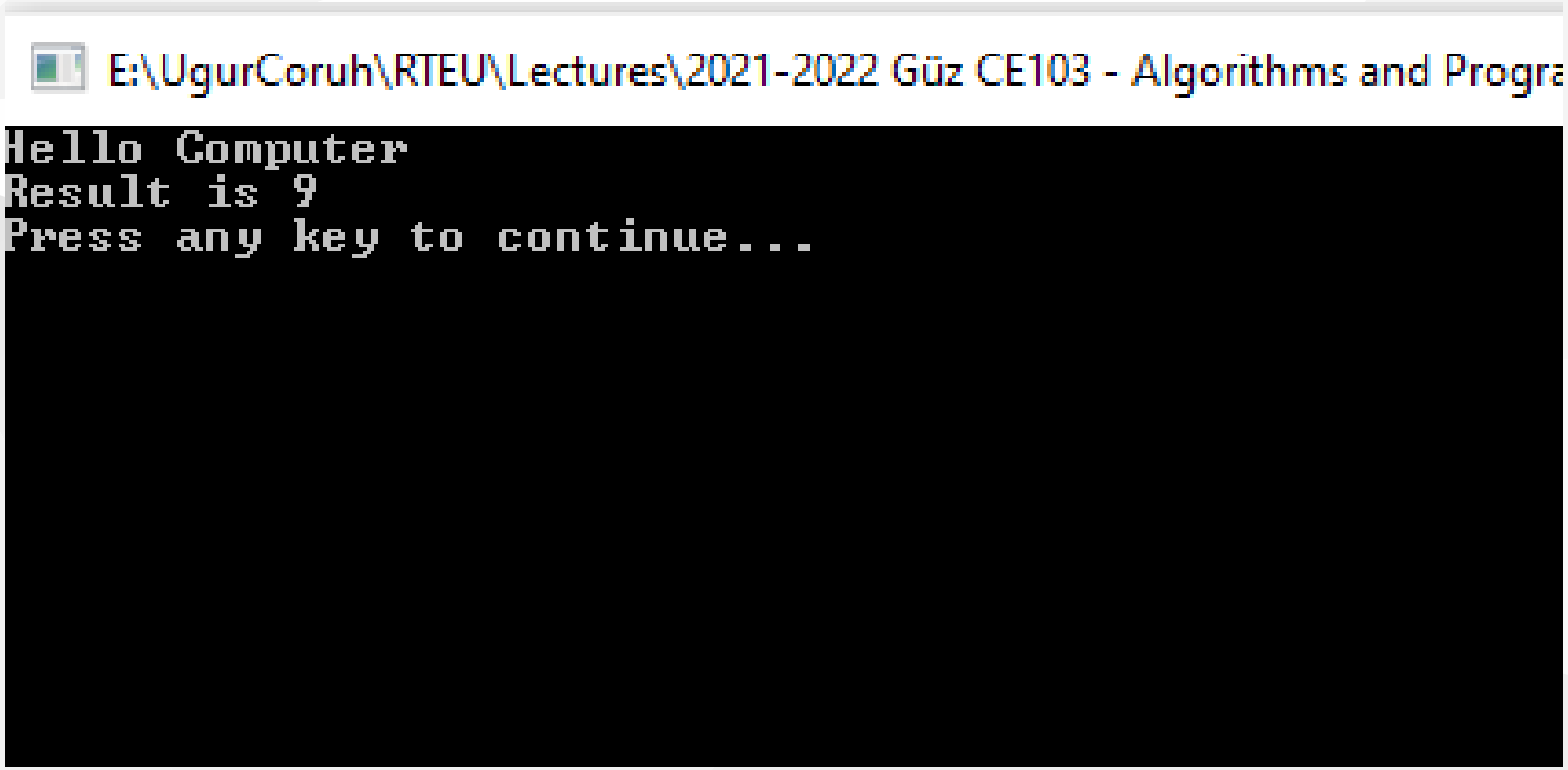
Shared Library Development - (VS C Static Library)-44

current source code will be like that nothing changed

```c
#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;
}
```

Shared Library Development - (VS C Static Library)-45

- and output will be as follow

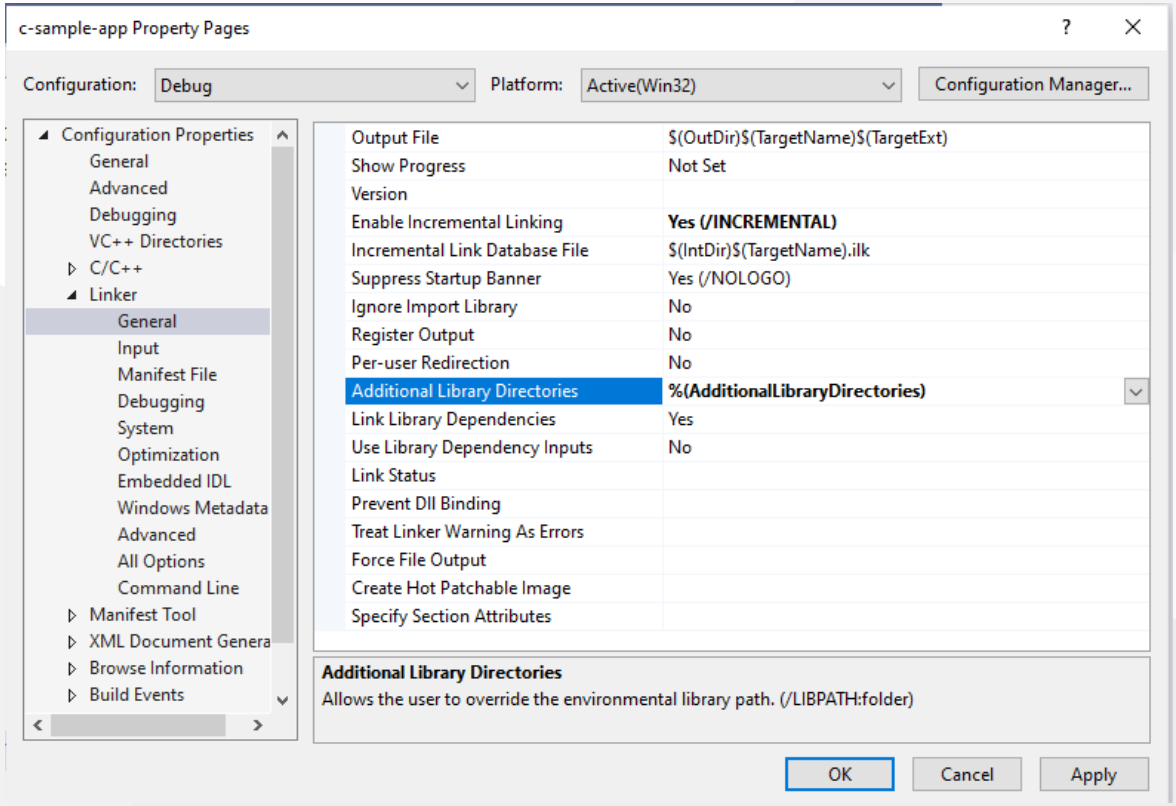


Shared Library Development - (VS C Static Library)-46

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

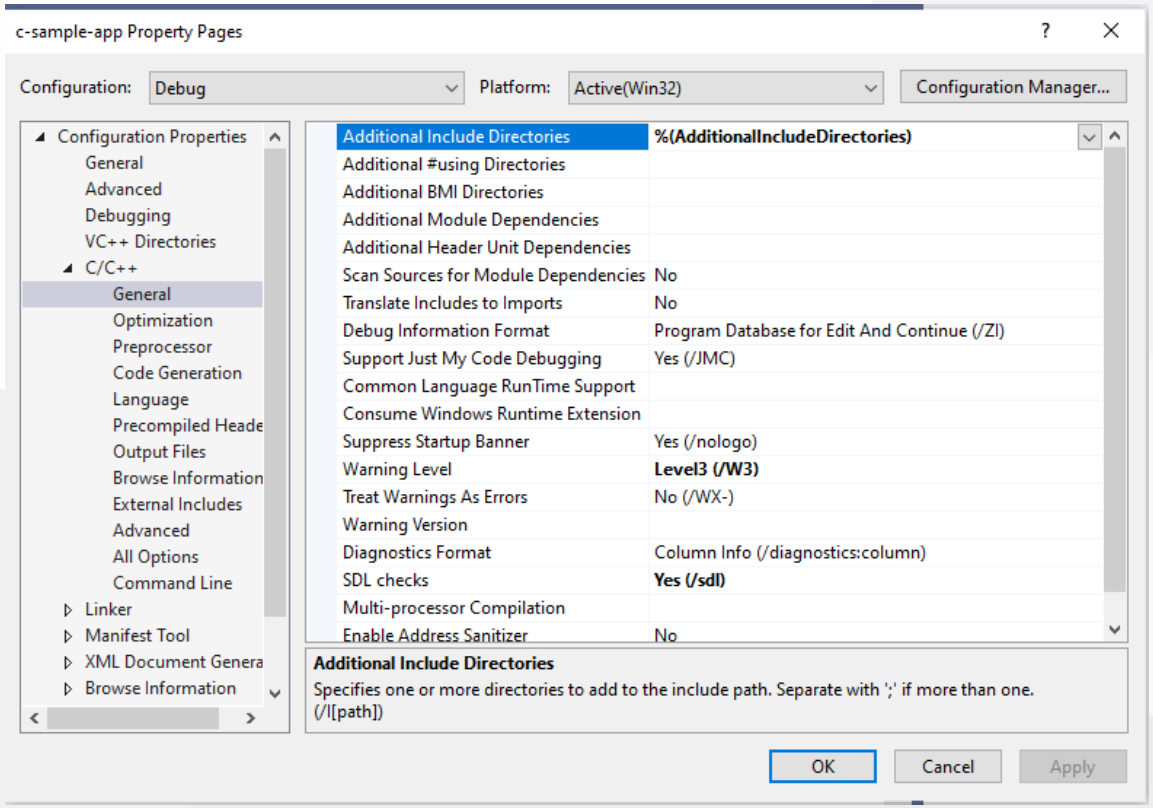
We will remove all library related settings from configurations and we will write them in source code

Clear linker->general->additional library directories



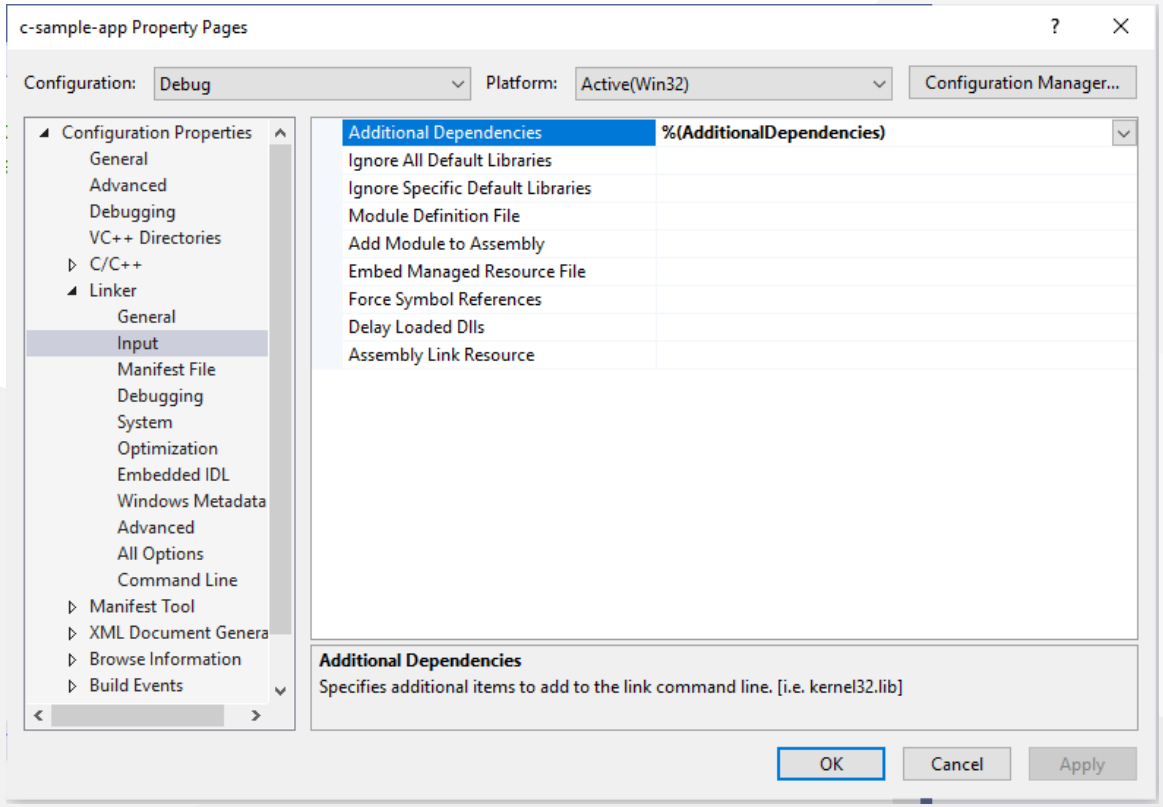
Shared Library Development - (VS C Static Library)-47

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



Shared Library Development - (VS C Static Library)-48

Clear Linker->Input->Additional Dependencies



Shared Library Development - (VS C Static Library)-49

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.

Shared Library Development

C++ Programming (Static Library)

Visual Studio Community Edition

Shared Library Development - (VS Cpp Static Library)-1

- 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.

Shared Library Development - (VS Cpp Static Library)-2

- 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;

}
```

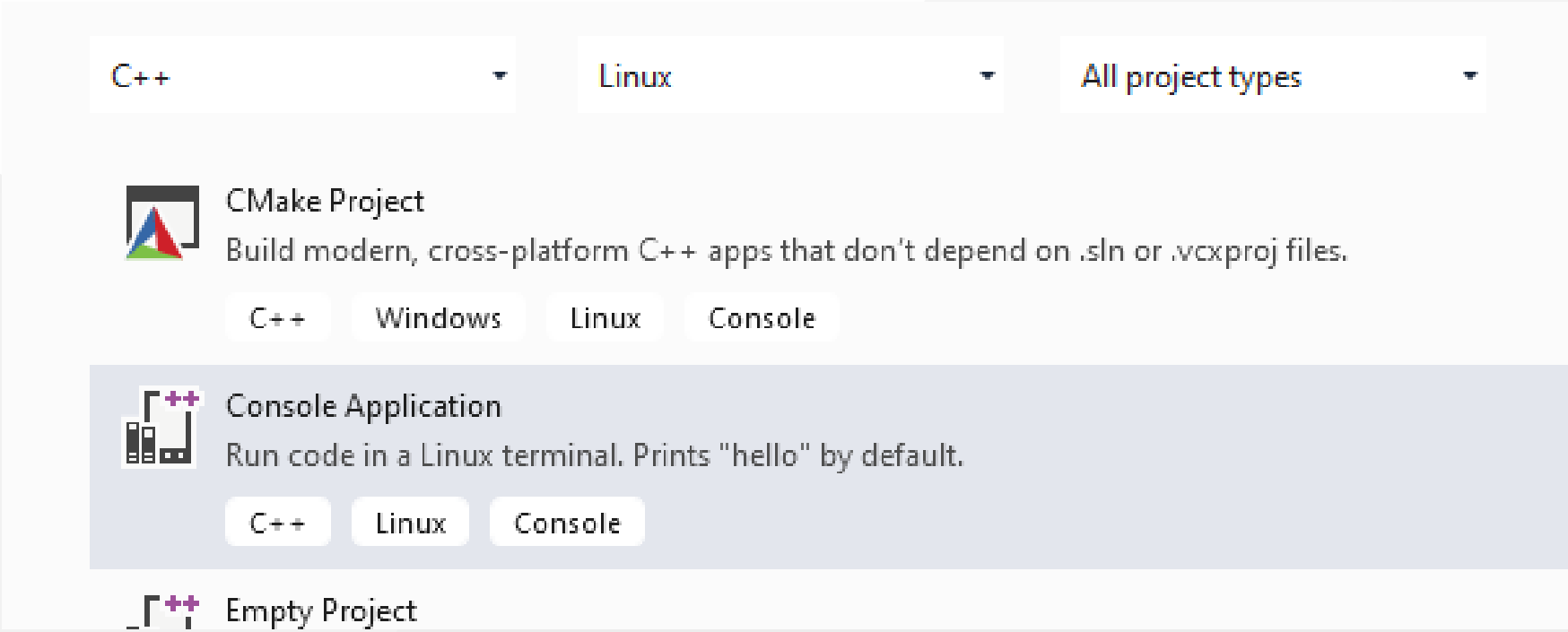
Shared Library Development

C++ Programming (Static Library)

Visual Studio Community Edition WSL Option

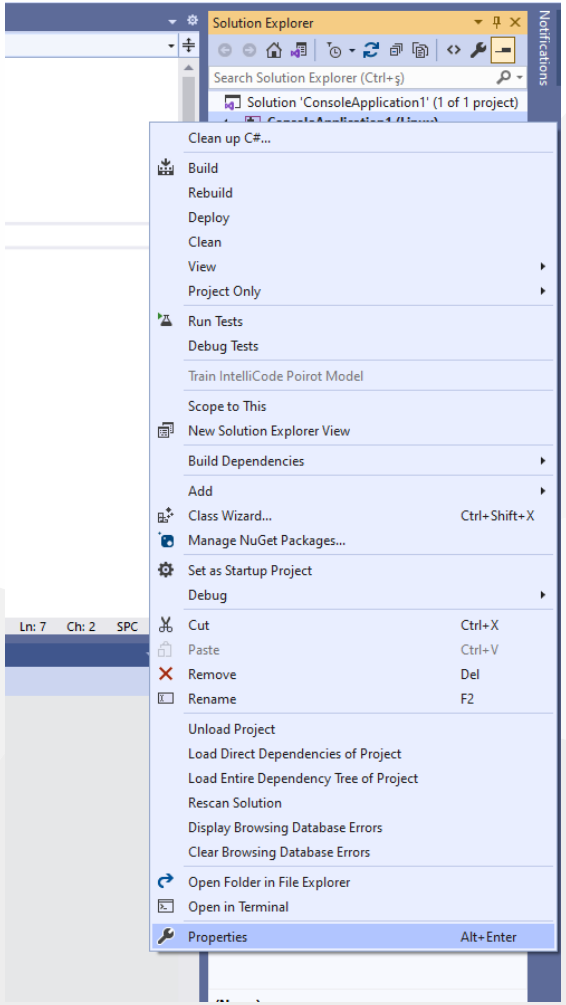
Shared Library Development - (VS Cpp WSL Static Library)-1

- Install WSL2
  - GitHub - ucoruh/ns3-wsl-win10-setup: ns3 windows 10 WSL2 setup and usage
- Create a Linux project



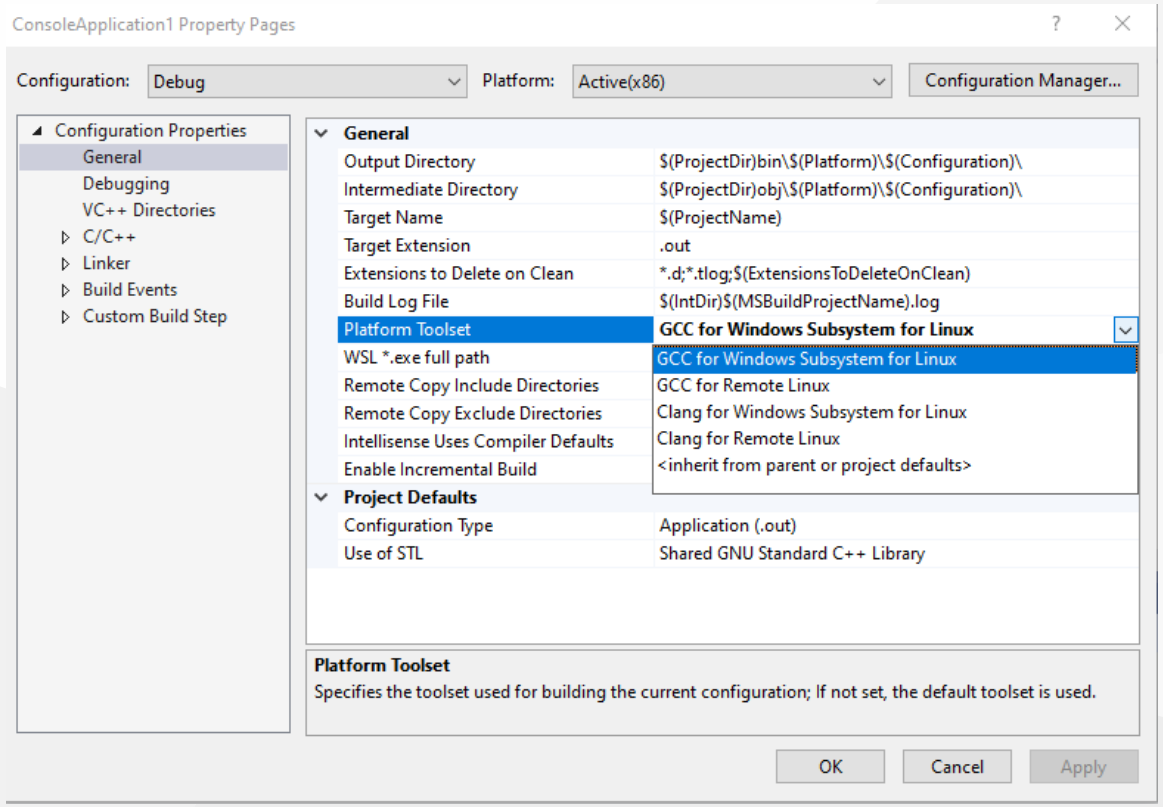
Shared Library Development - (VS Cpp WSL Static Library)-2

- Configure Platform Toolset to WSL



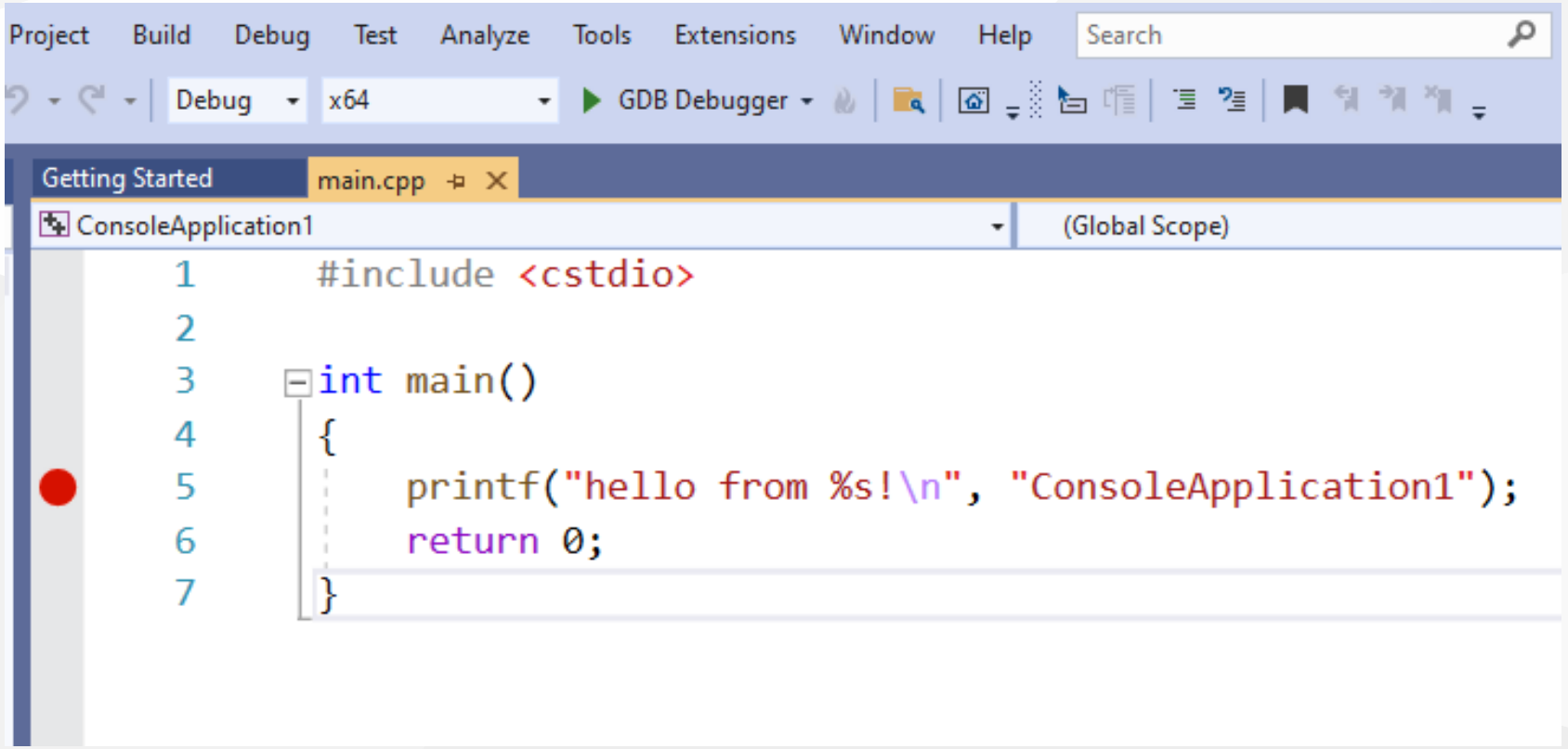
Shared Library Development - (VS Cpp WSL Static Library)-3

- Select GCC for Windows Subsystem for Linux



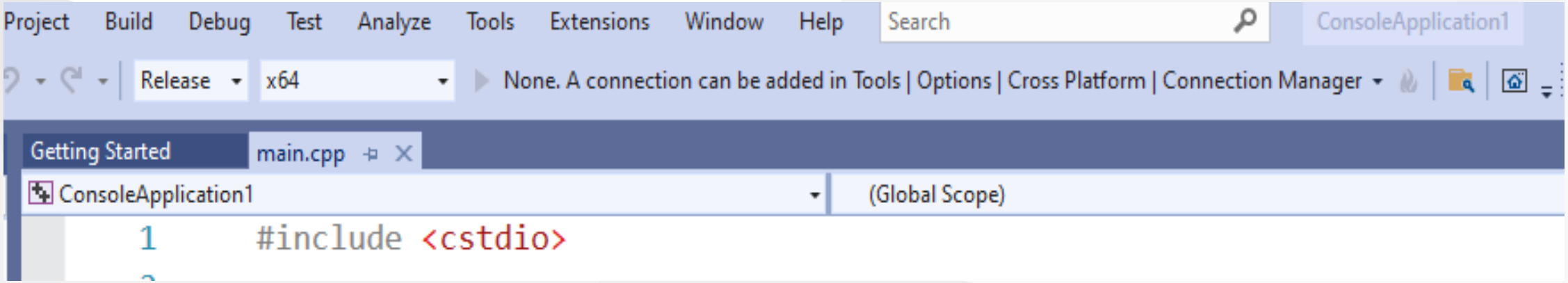
Shared Library Development - (VS Cpp WSL Static Library)-4

Put a breakpoint and run debugger



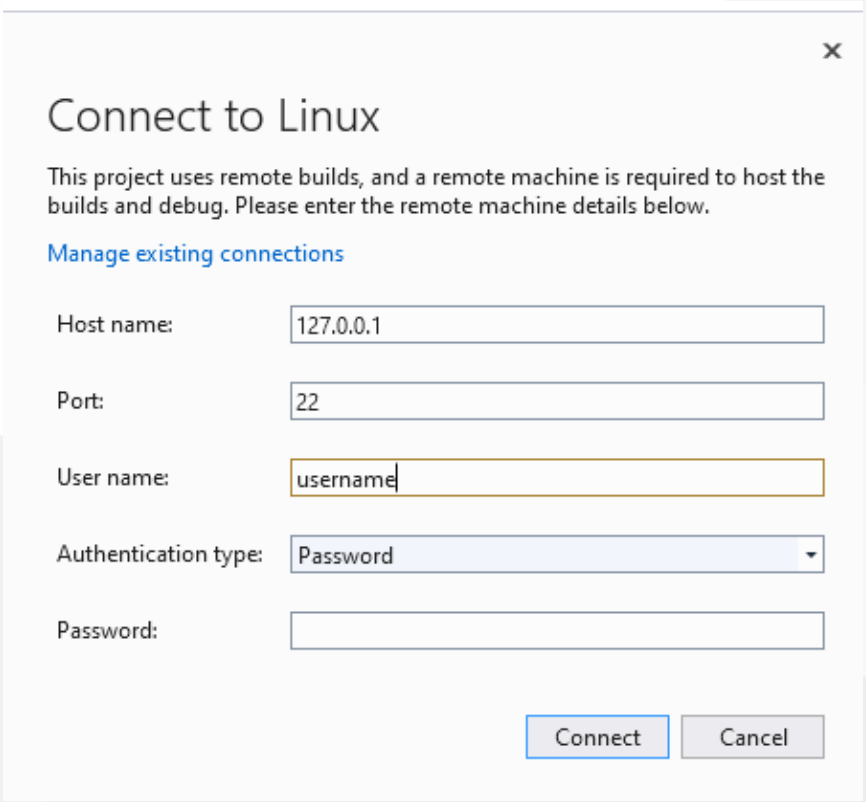
Shared Library Development - (VS Cpp WSL Static Library)-5

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.



Shared Library Development - (VS Cpp WSL Static Library)-6

- Configure SSH parameters



Shared Library Development - (VS Cpp WSL Static Library)-7

- so you have to complete the following steps.
- C/C++ Remote Linux Option over SSH
  - Enable SSH
    - SSH on Windows Subsystem for Linux (WSL) | Illuminia Studios
  - Connect to Remote WSL Environment
    - Bağlan hedef Linux sisteminize Visual Studio | Microsoft Docs

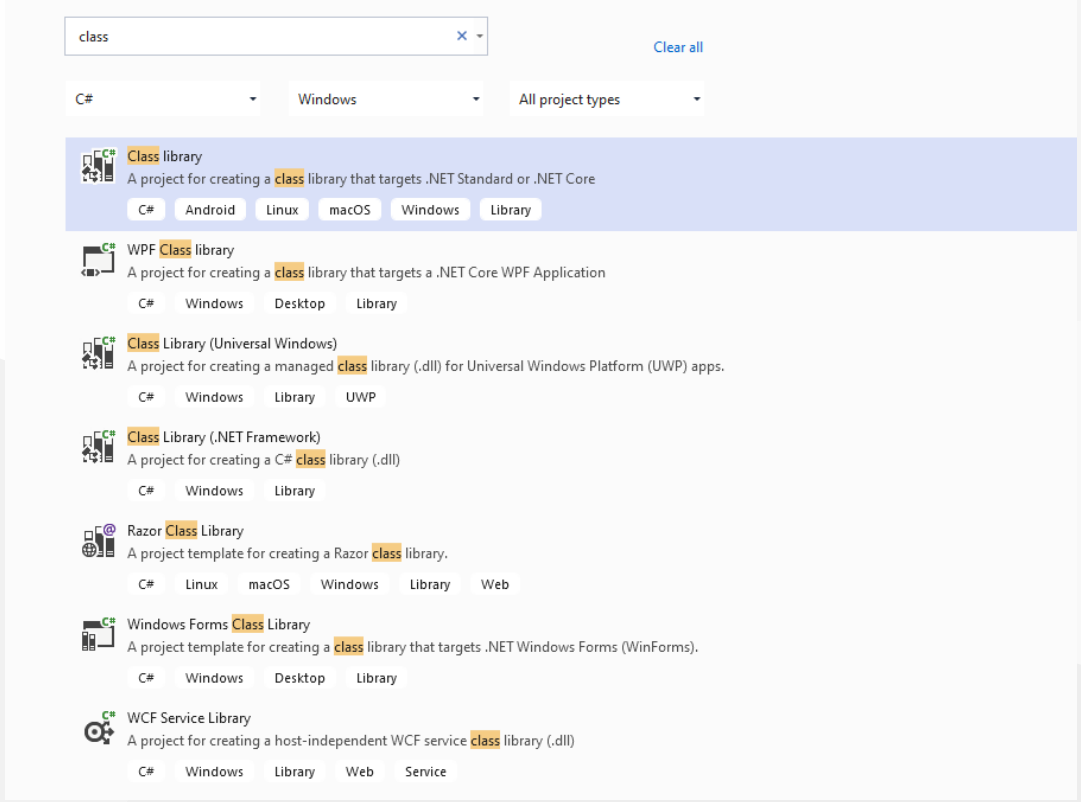
Shared Library Development

C# Programming (Dinamik Library)

Visual Studio Community Edition

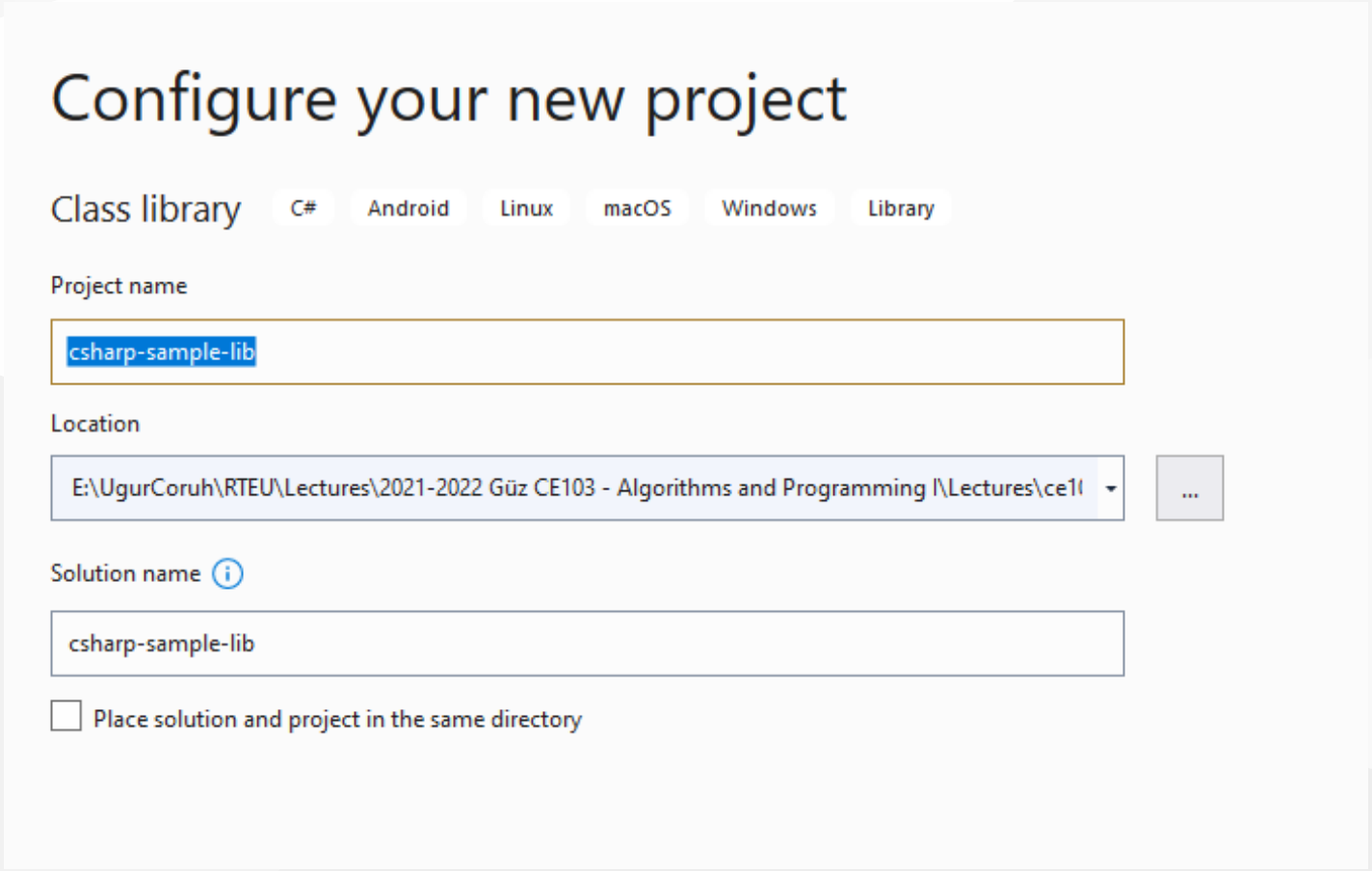
Shared Library Development - (VS Csharp Dynamic Library)-1

- 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



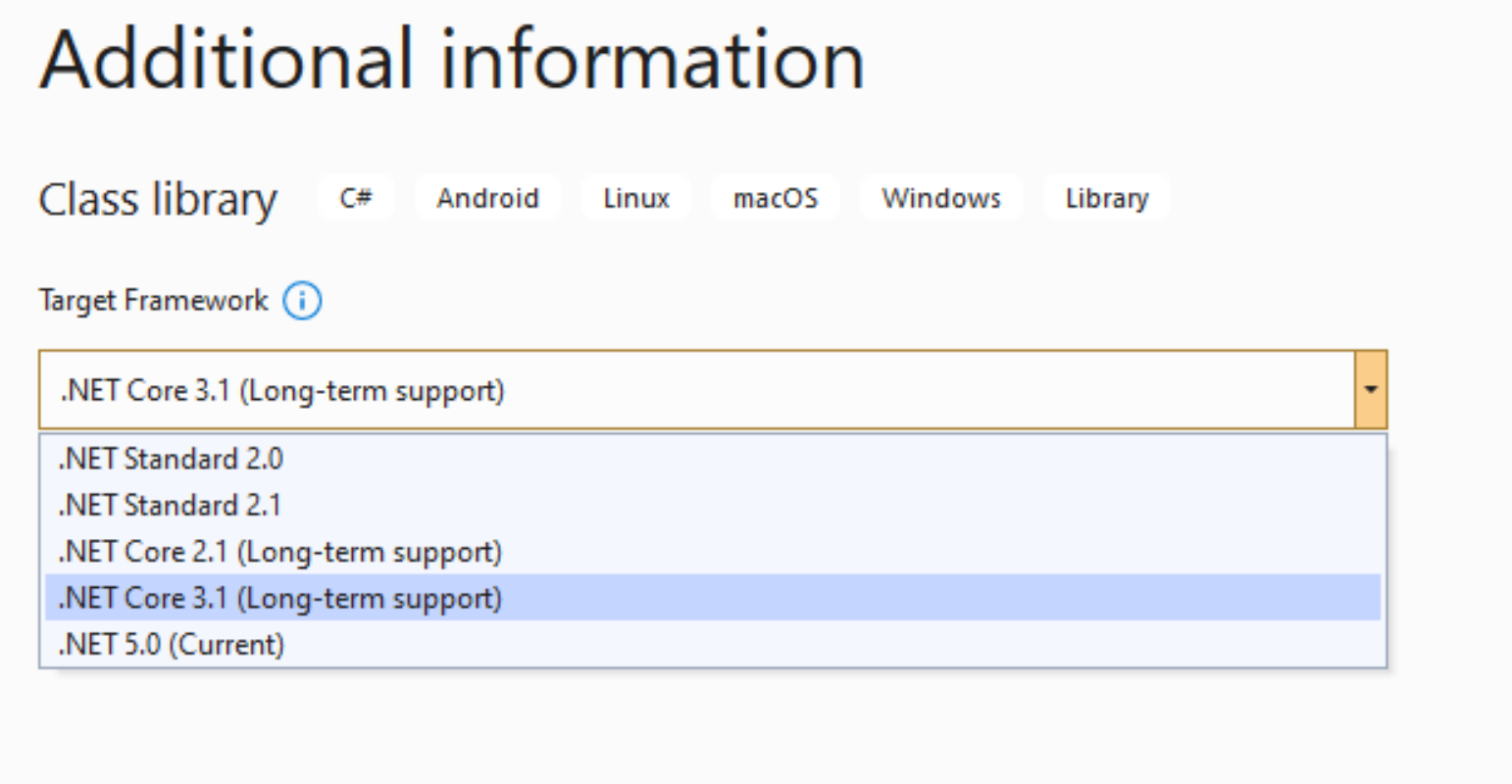
Shared Library Development - (VS Csharp Dynamic Library)-2

- There is no static library option



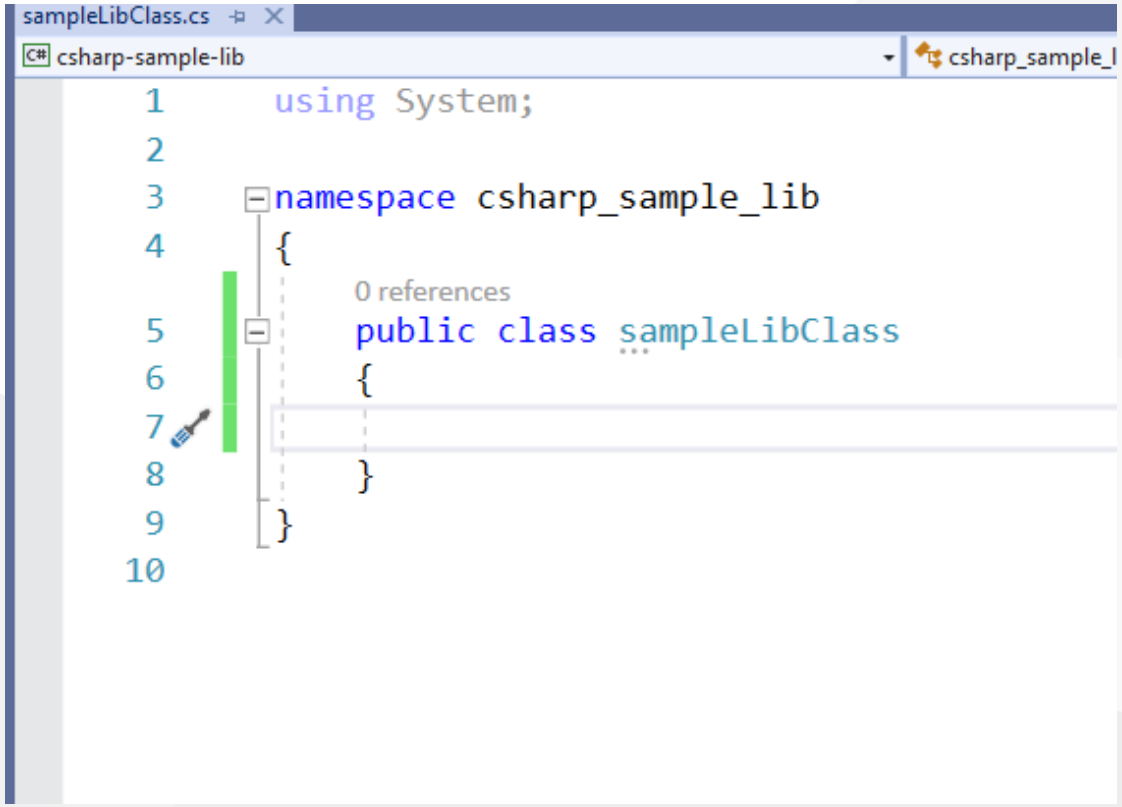
Shared Library Development - (VS Csharp Dynamic Library)-3

- We will select .Net Core 3.1



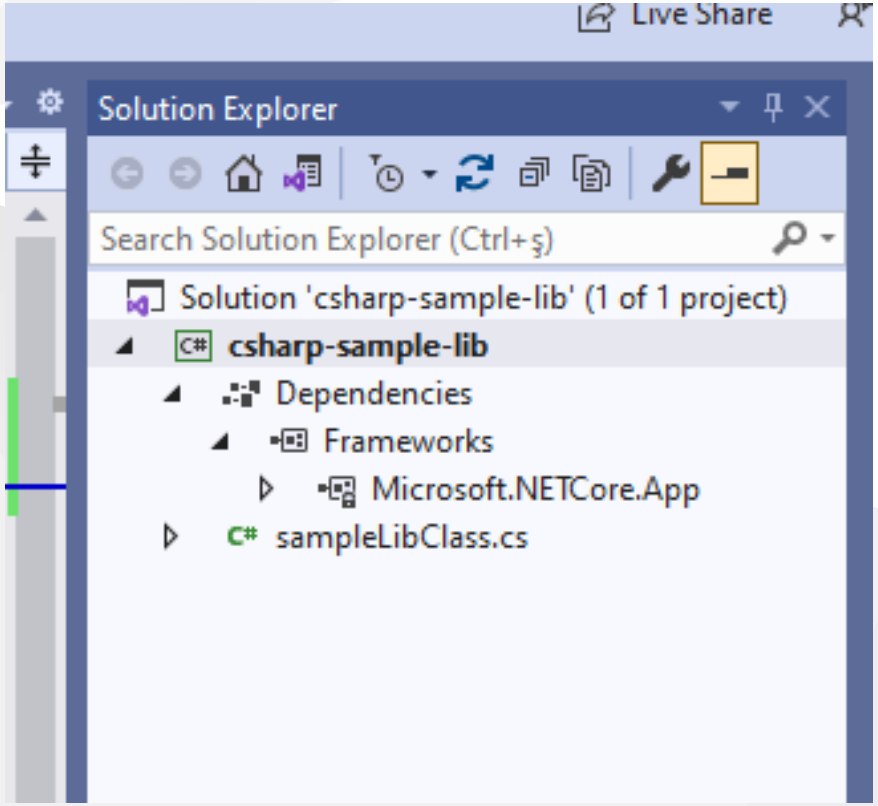
Shared Library Development - (VS Csharp Dynamic Library)-4

- You will have default empty class library file



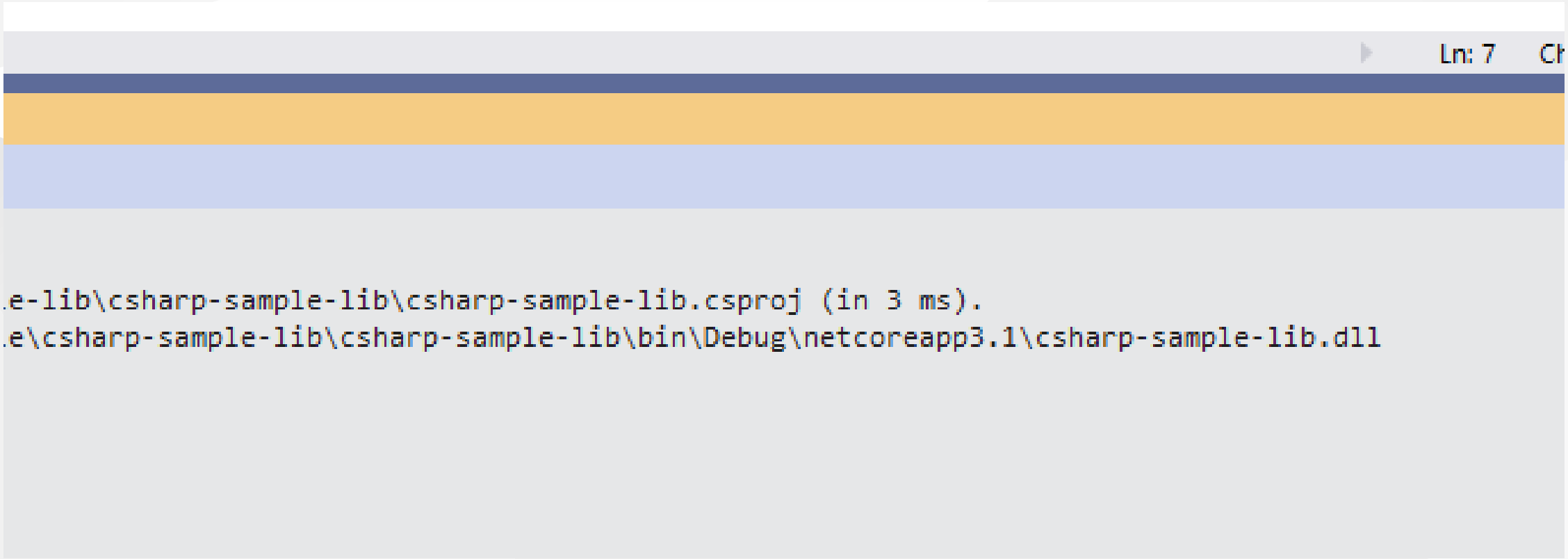
Shared Library Development - (VS Csharp Dynamic Library)-5

- In the project you can see .NETcore reference



Shared Library Development - (VS Csharp Dynamic Library)-6

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

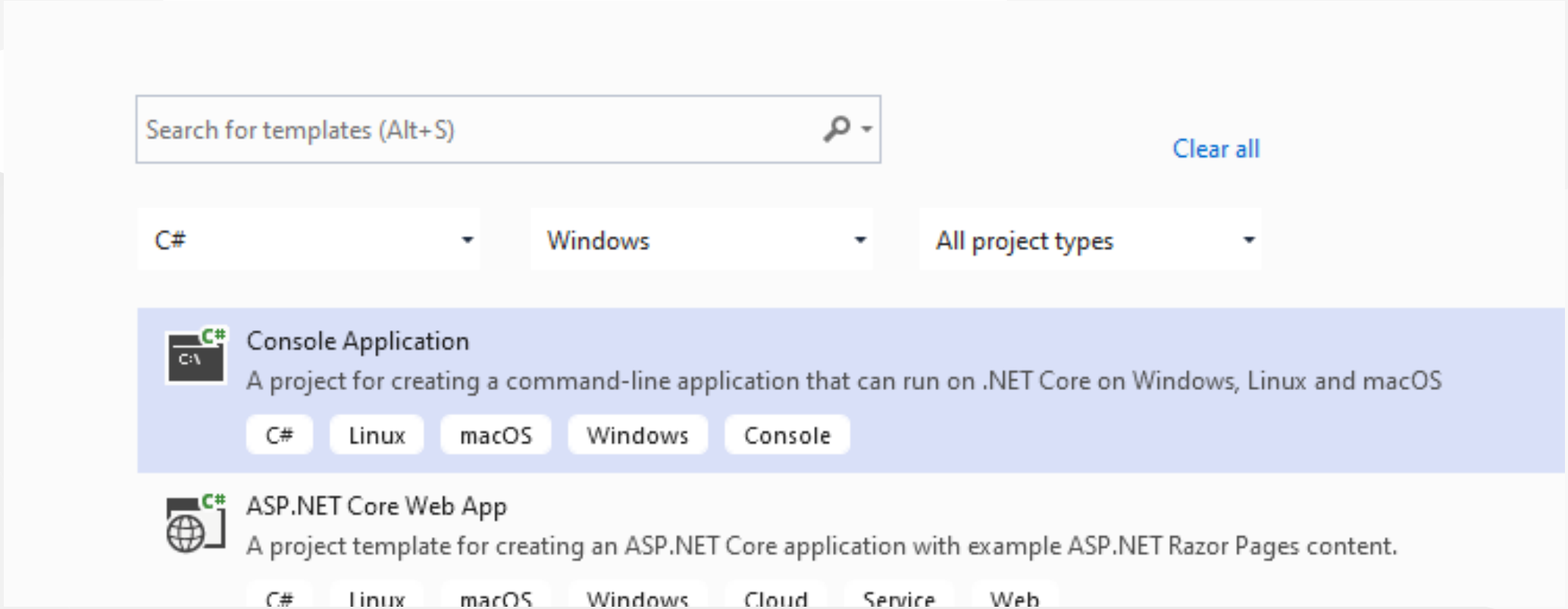


Shared Library Development - (VS Csharp Dynamic Library)-7

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

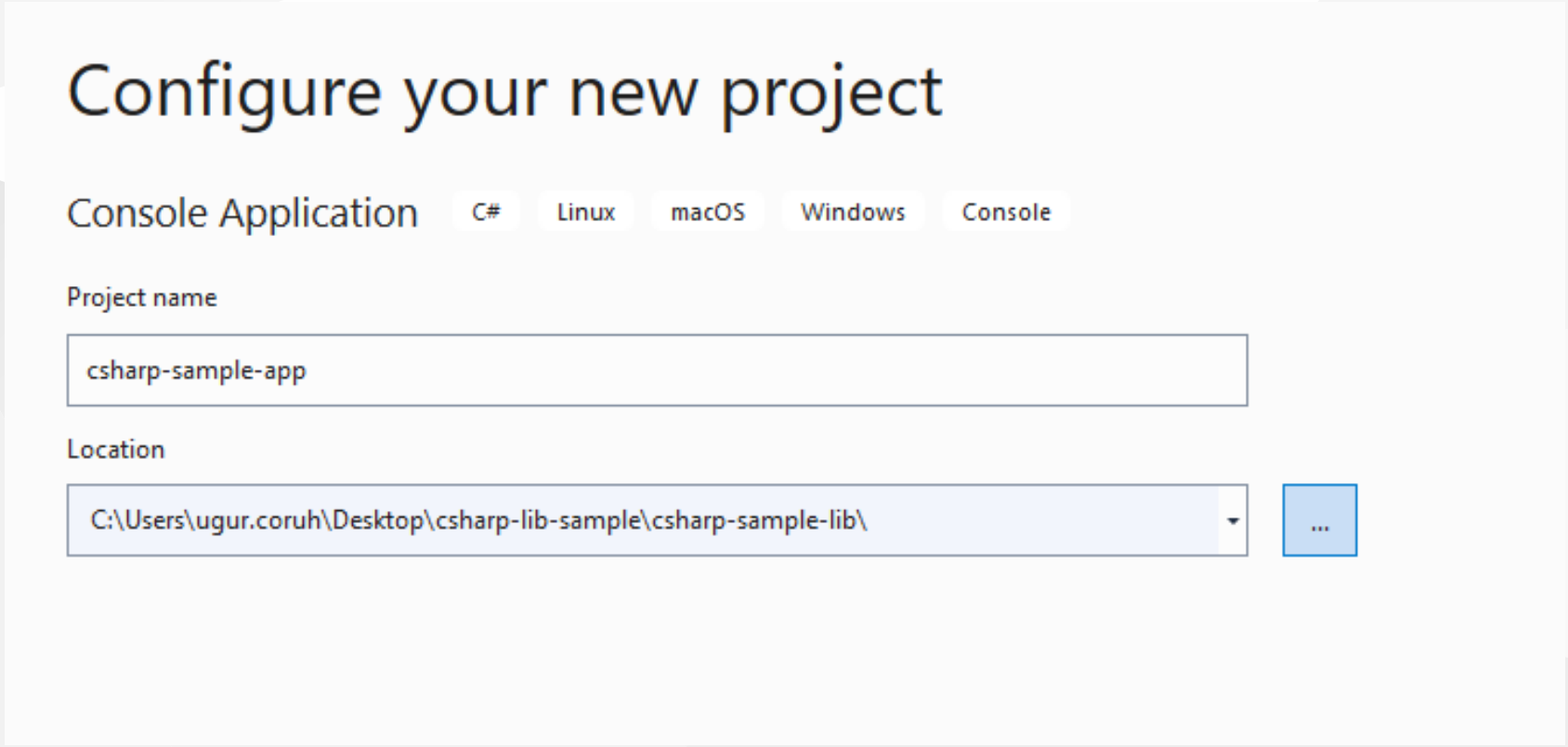
Shared Library Development - (VS Csharp Dynamic Library)-8

- Select New Project



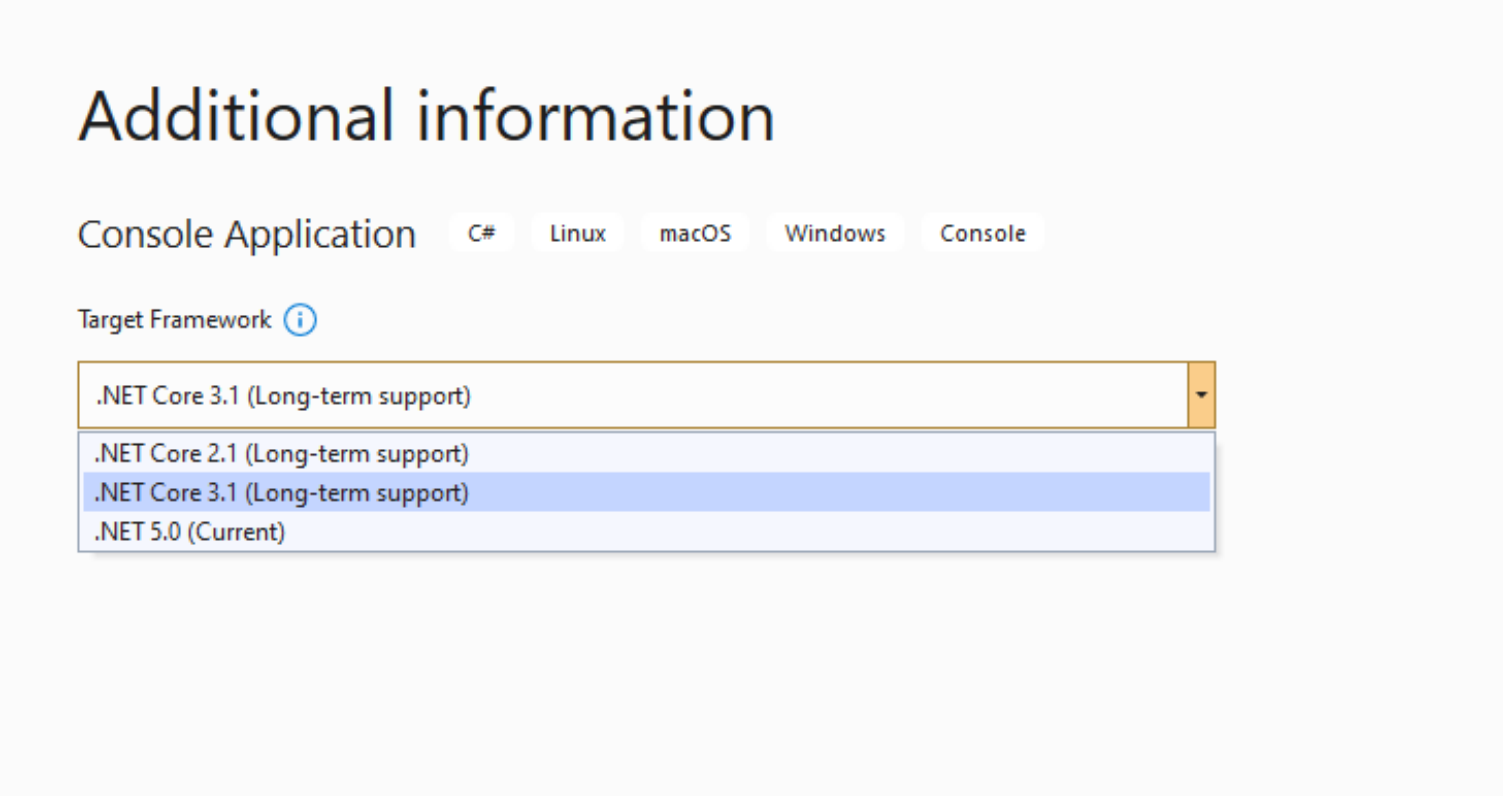
Shared Library Development - (VS Csharp Dynamic Library)-9

- Set project name



Shared Library Development - (VS Csharp Dynamic Library)-10

- Select .NETCore framework



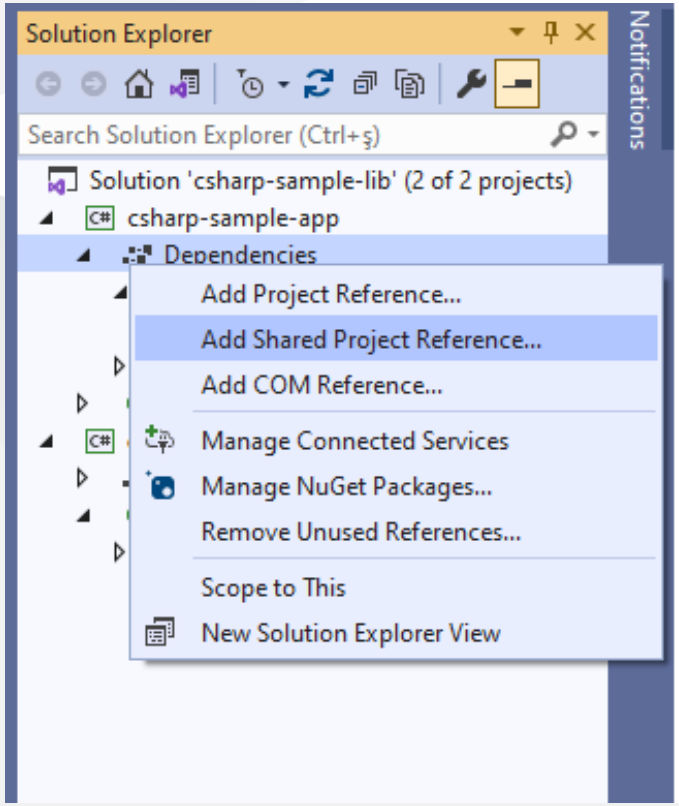
Shared Library Development - (VS Csharp Dynamic Library)-11

- 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!");
        }
    }
}
```

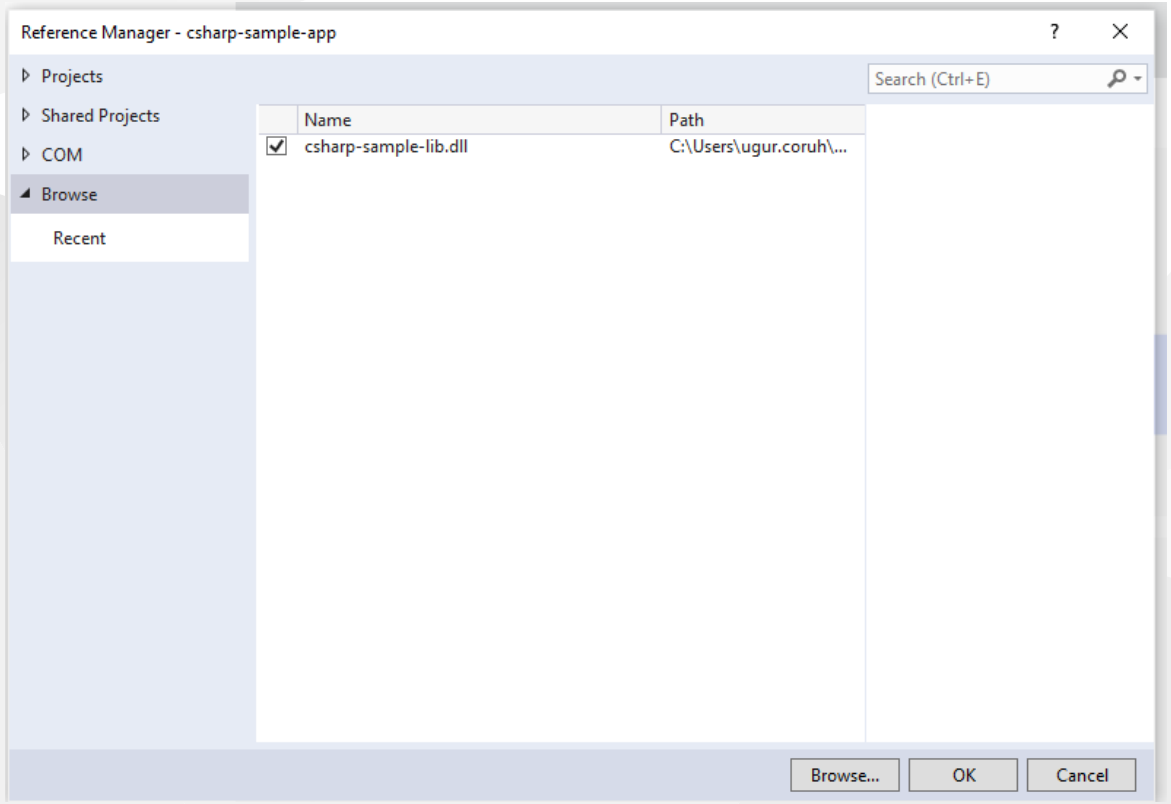
Shared Library Development - (VS Csharp Dynamic Library)-12

- Now we can link projects with adding references open reference section



Shared Library Development - (VS Csharp Dynamic Library)-13

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



Shared Library Development - (VS Csharp Dynamic Library)-14

- now we can update our library code and use it in console application
- copy following sample to sampleLibClass file in the library

Shared Library Development - (VS Csharp Dynamic Library)-15

```
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;
        }
    }
}
```

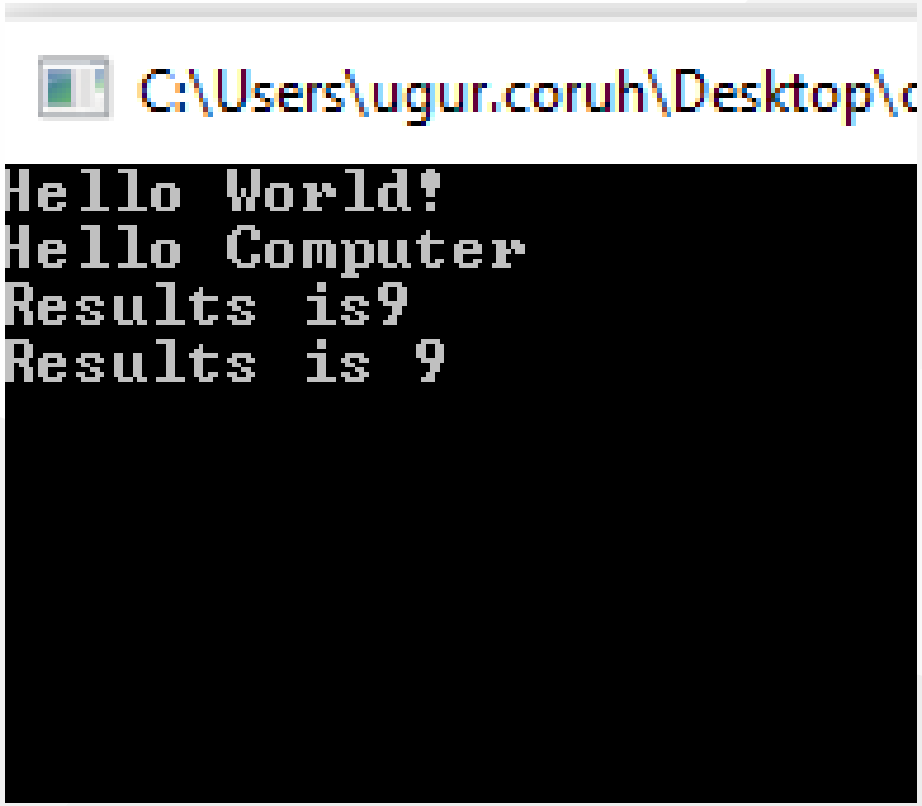
Shared Library Development - (VS Csharp Dynamic Library)-16

- 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();
        }
    }
}
```

Shared Library Development - (VS Csharp Dynamic Library)-17

- You will see following output that mean we called DLL functions



Shared Library Development - (VS Csharp Dynamic Library)-18

- Also we can publish this console application with dll for linux environment or others
- for linux environment we should install .NETCore

Shared Library Development - (VS Csharp Dynamic Library)-19

- 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

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
```

Shared Library Development - (VS Csharp Dynamic Library)-20

Step 2 – Installing Dotnet Core SDK

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

Shared Library Development - (VS Csharp Dynamic Library)-21

Step 3 – Install Dotnet Core Runtime Only

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

```
sudo apt update
```

Shared Library Development - (VS Csharp Dynamic Library)-22

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

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

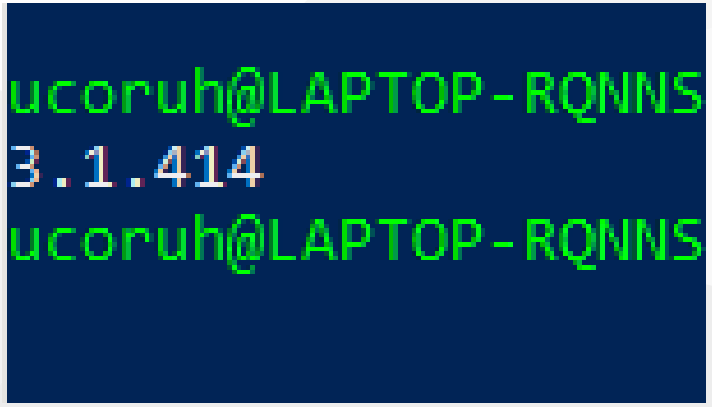
Press “y” for any input prompted by the installer.

Shared Library Development - (VS Csharp Dynamic Library)-23

Step 4 – (Optional) Check .NET Core Version

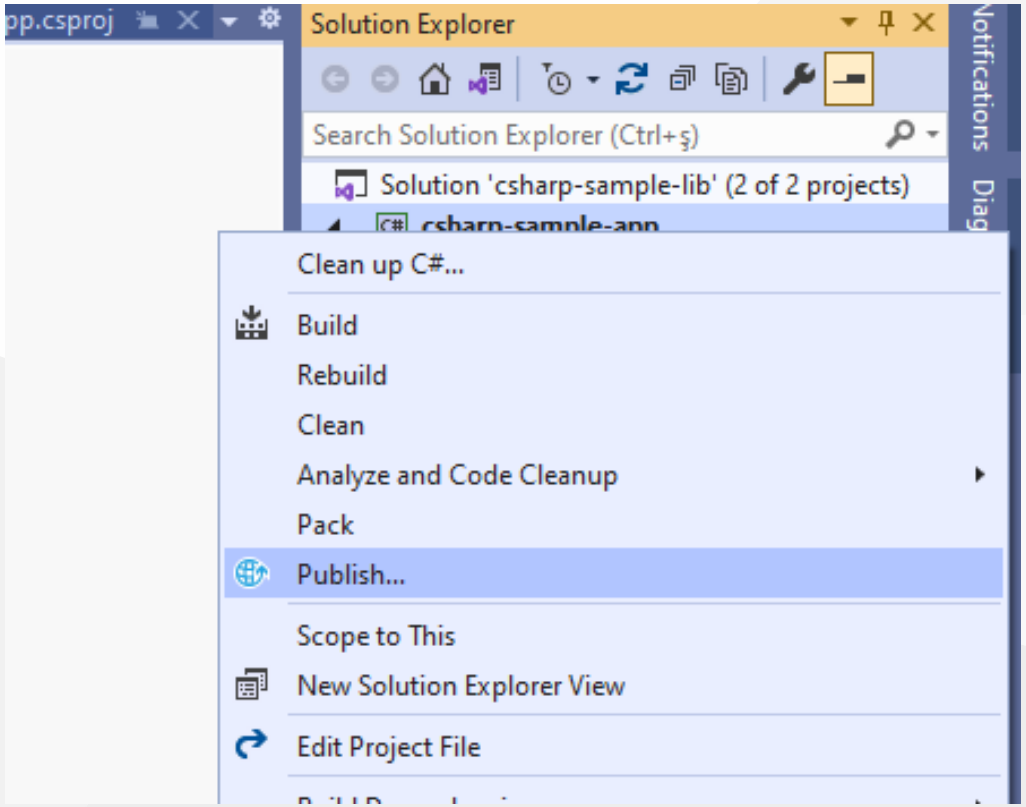
You can use dotnet command line utility to check installed version of .NET Core on your system. To check dotnet version, type:

```
dotnet --version
```



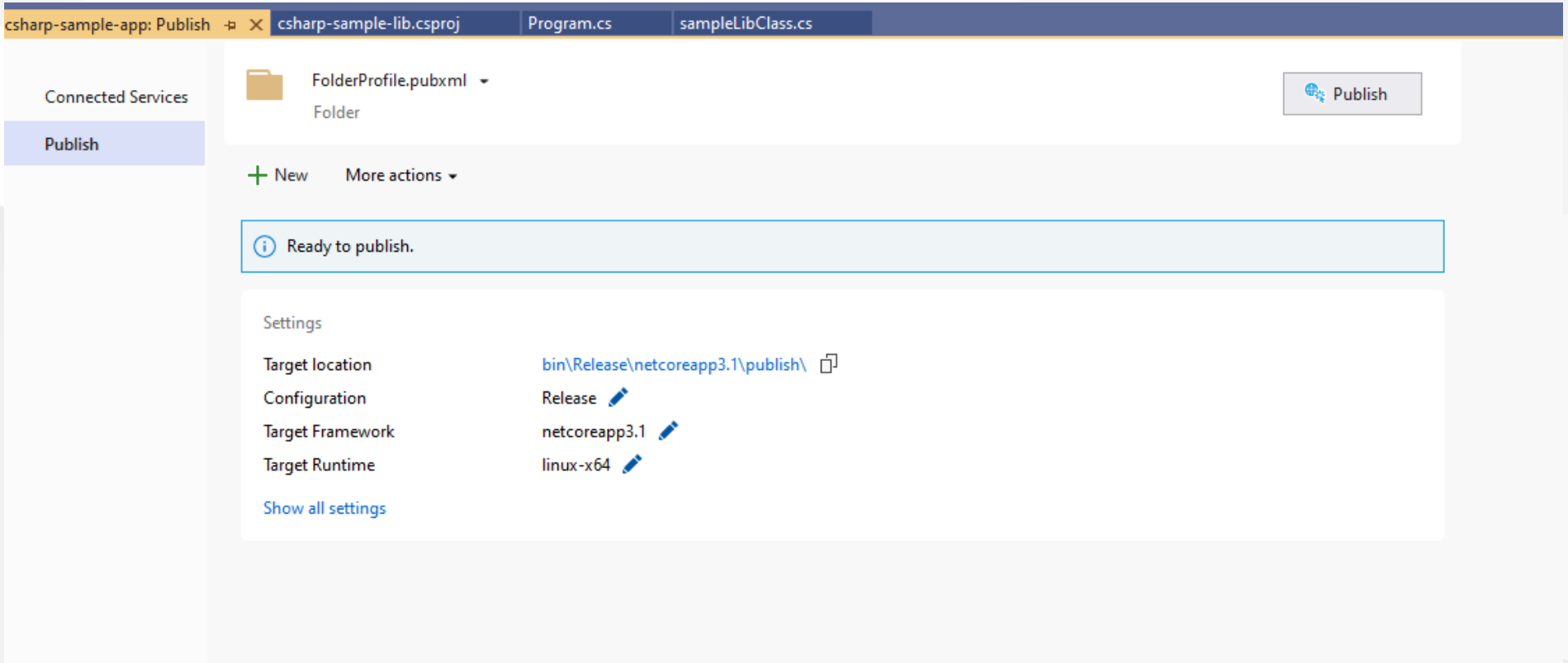
Shared Library Development - (VS Csharp Dynamic Library)-24

- Now we will publish our application as single executable
- Open publish menu



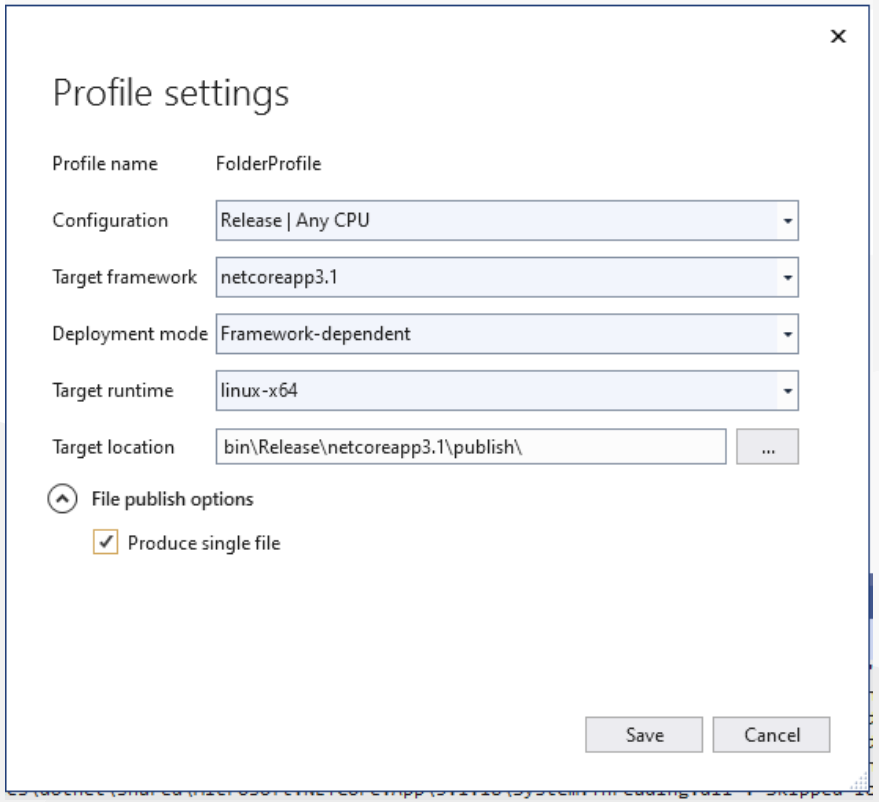
Shared Library Development - (VS Csharp Dynamic Library)-25

- Select netcoreapp3.1 and Release for linux-x64



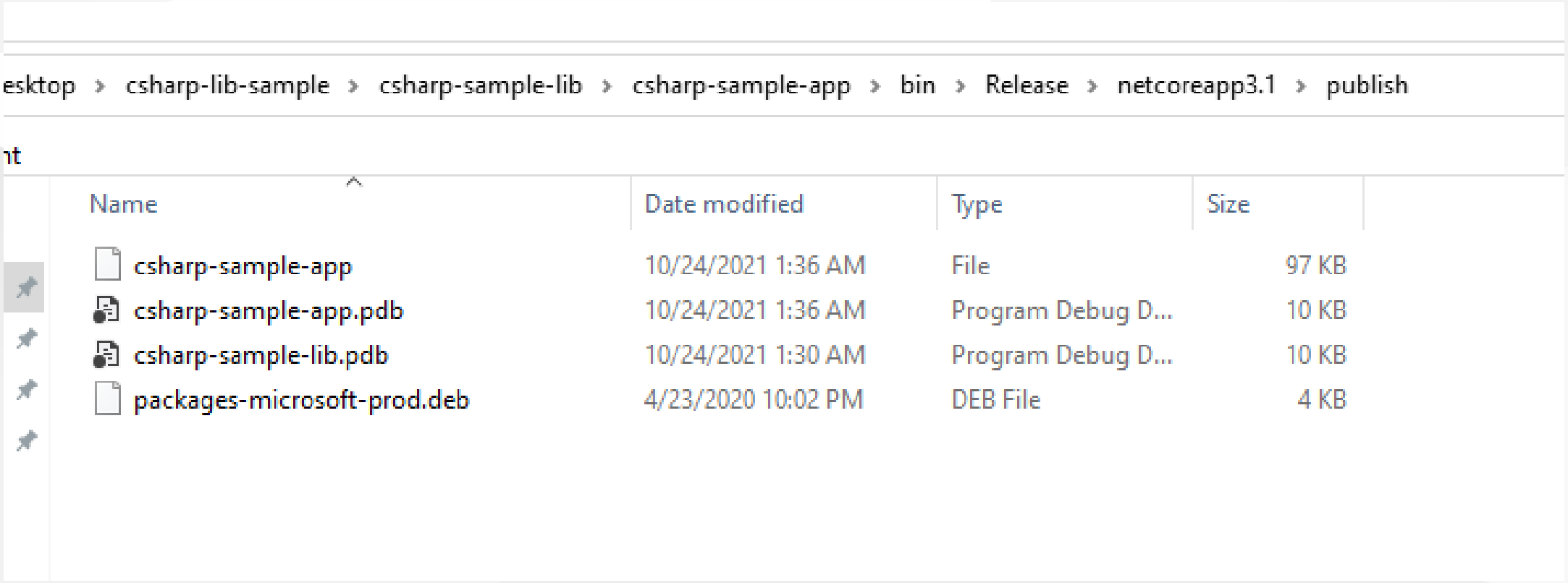
Shared Library Development - (VS Csharp Dynamic Library)-26

- Select produce single file



Shared Library Development - (VS Csharp Dynamic Library)-27

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



Shared Library Development - (VS Csharp Dynamic Library)-28

- Open WSL and enter the path where this folder located
- And run application as follow

```
Processing triggers for man-db (2.9.1-1) ...
ucoruh@LAPTOP-RQNNS9IG:/mnt/c/Users/ugur.coruh/Desktop/csharp-lib-sample/csharp-sample-lib/csharp-sample-app/bin/Release/netcoreapp3.1/publish$ dotnet --version
3.1.414
ucoruh@LAPTOP-RQNNS9IG:/mnt/c/Users/ugur.coruh/Desktop/csharp-lib-sample/csharp-sample-lib/csharp-sample-app/bin/Release/netcoreapp3.1/publish$ ./
csharp-sample-app          csharp-sample-app.pdb       csharp-sample-lib.pdb       packages-microsoft-prod.deb
ucoruh@LAPTOP-RQNNS9IG:/mnt/c/Users/ugur.coruh/Desktop/csharp-lib-sample/csharp-sample-lib/csharp-sample-app/bin/Release/netcoreapp3.1/publish$ ./csharp-sample-app
Hello World!
Hello Computer
Results is9
Results is 9
```

Shared Library Development - (VS Csharp Dynamic Library)-29

check dotnet --version and then run application

```
ublish$ dotnet --version

ublish$ ./

ublish$ ./csharp-sample-app
```

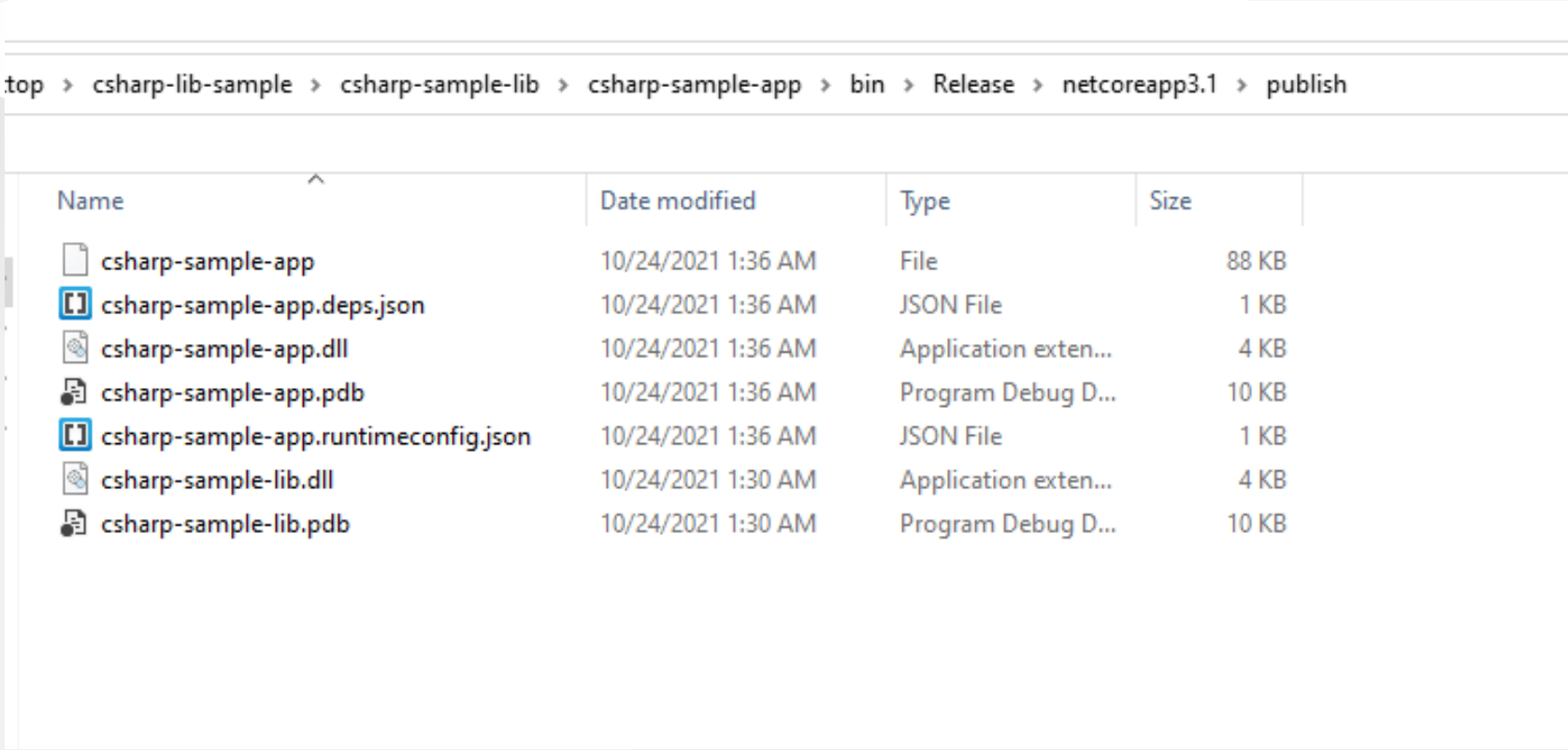
- you will see similar output

```
ucoruh@LAPTOP-RQNNS9IG:/mnt/c/
csharp-sample-app
ucoruh@LAPTOP-RQNNS9IG:/mnt/c/
Hello World!
Hello Computer
Results is9
Results is 9
```

Shared Library Development - (VS Csharp Dynamic Library)-30

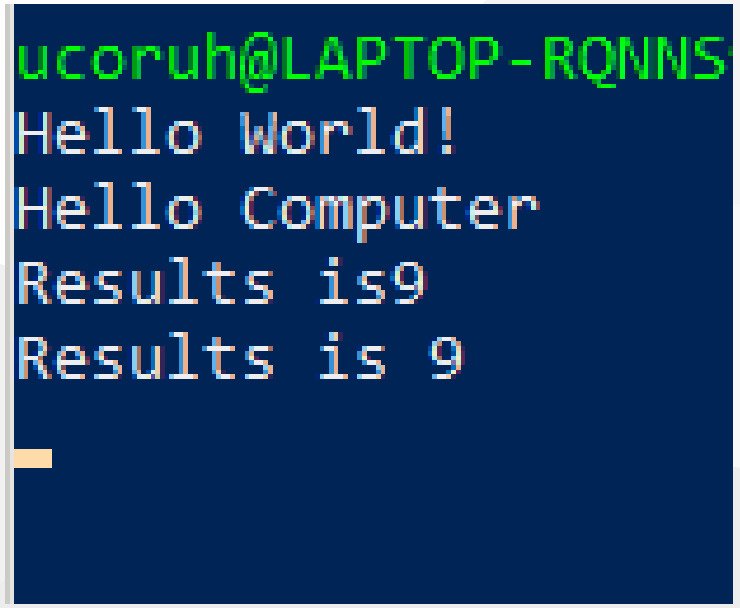
In this sample we created single application from settings lets try with shared library located option uncheck the "produce single file" option and publish again.

Then you will have the following outputs



Shared Library Development - (VS Csharp Dynamic Library)-31

- If you run csharp-sample-app
- you will have the same output



Shared Library Development

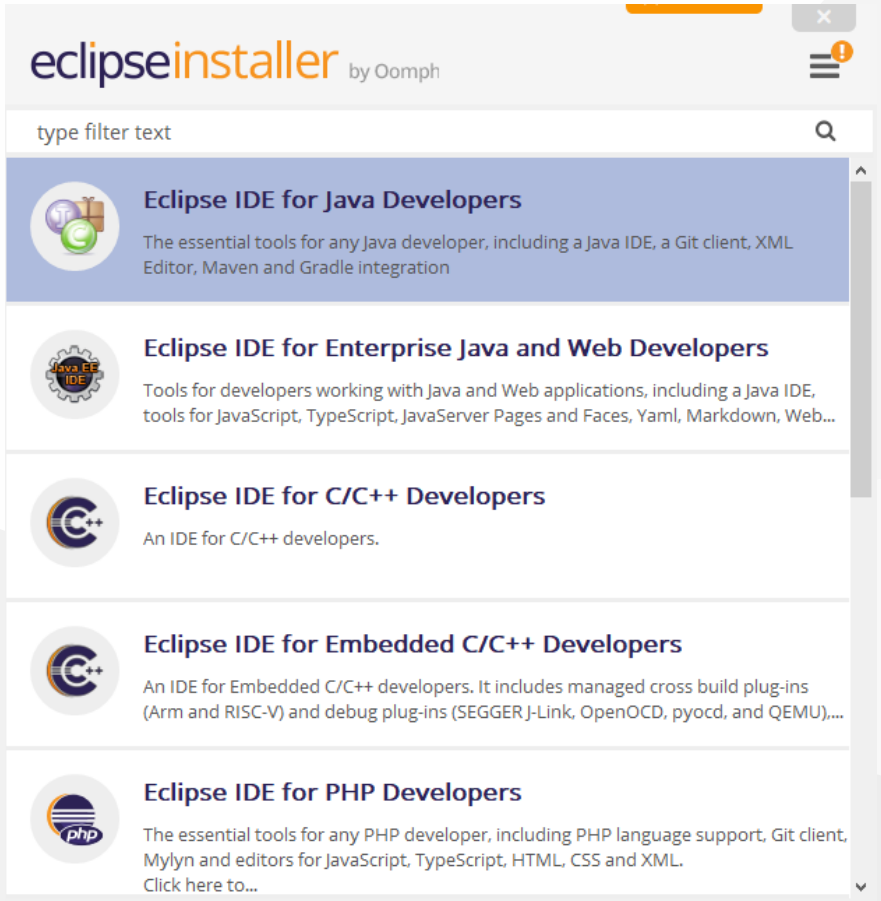
Java Programming

Eclipse IDE

Shared Library Development - (Eclipse Java Jar Library)-1

- You should download and install eclipse installer and then you should select Eclipse IDE for Java Developers
  - Eclipse Installer 2021-09 R | Eclipse Packages

Shared Library Development - (Eclipse Java Jar Library)-2

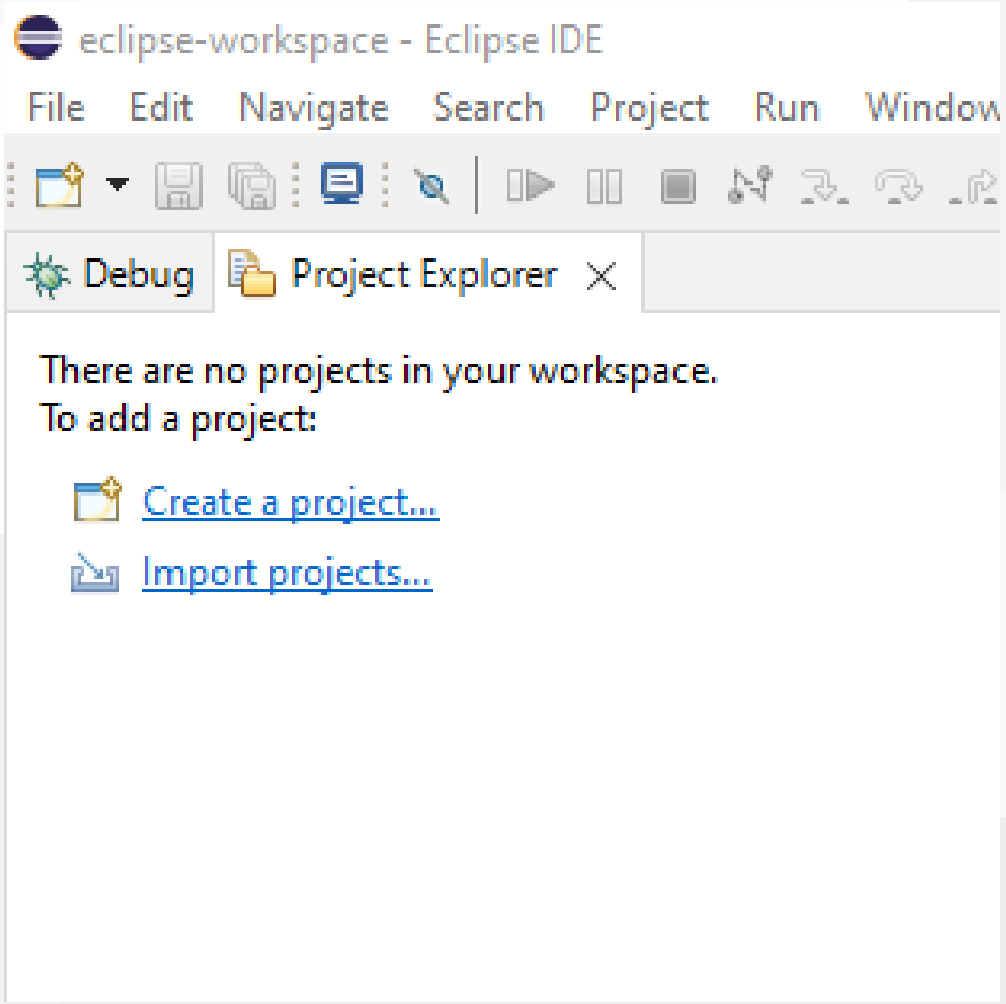


Shared Library Development - (Eclipse Java Jar Library)-3

Shared Library Development - (Eclipse Java Jar Library)-4

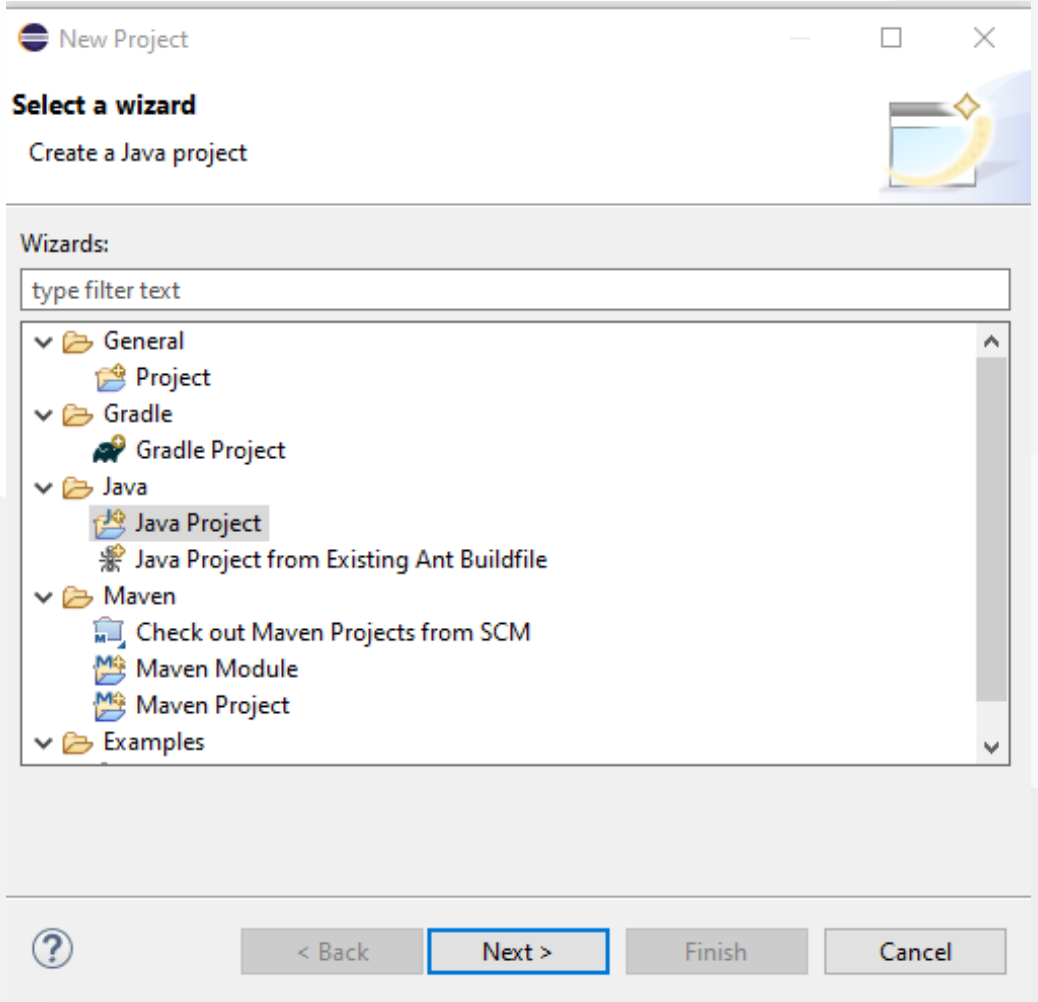
Shared Library Development - (Eclipse Java Jar Library)-5

- select create a project



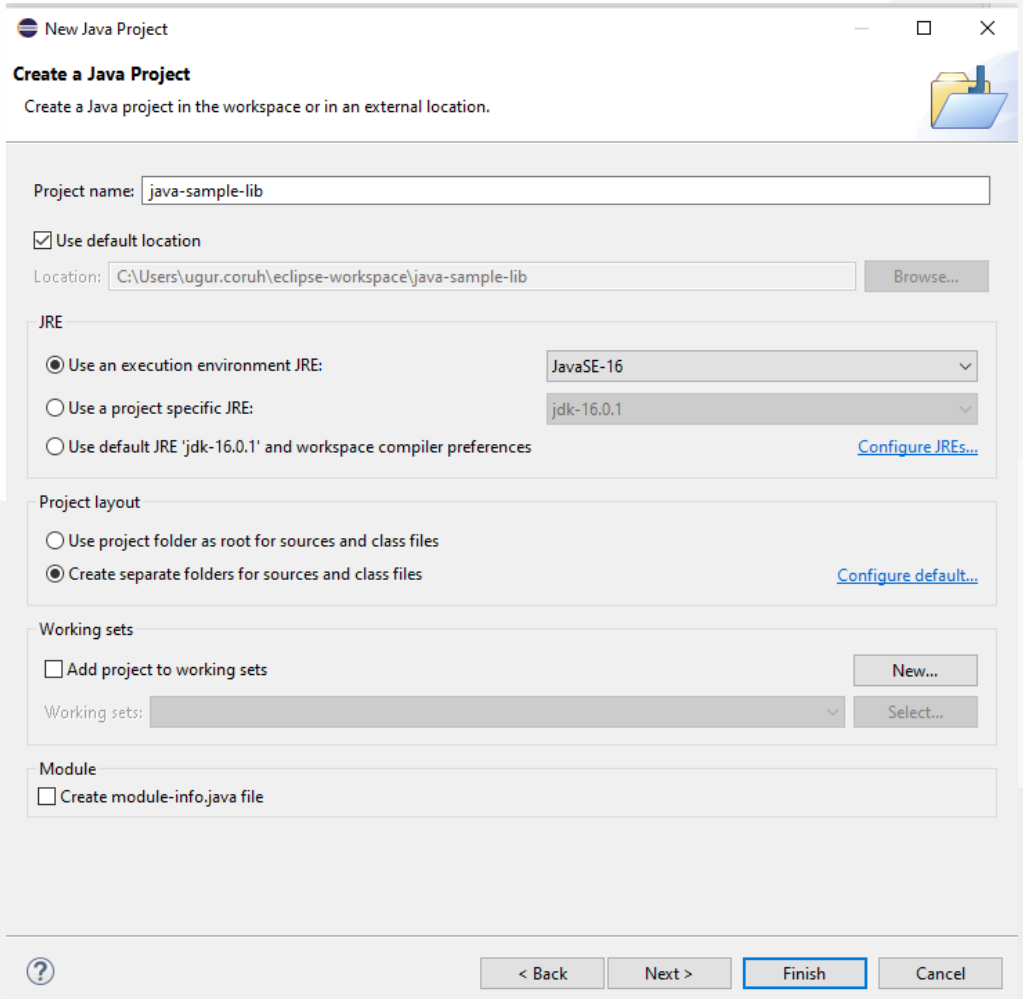
Shared Library Development - (Eclipse Java Jar Library)-6

select java project



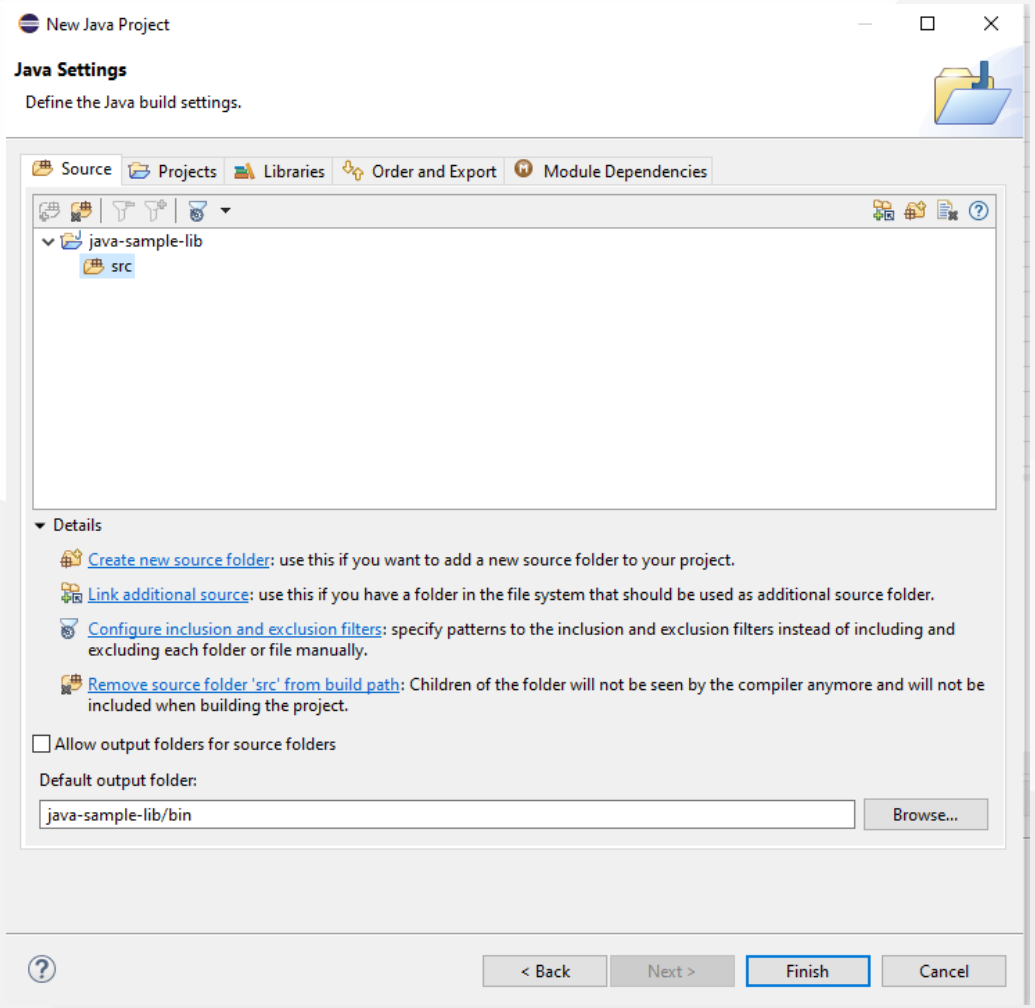
Shared Library Development - (Eclipse Java Jar Library)-7

- give project name



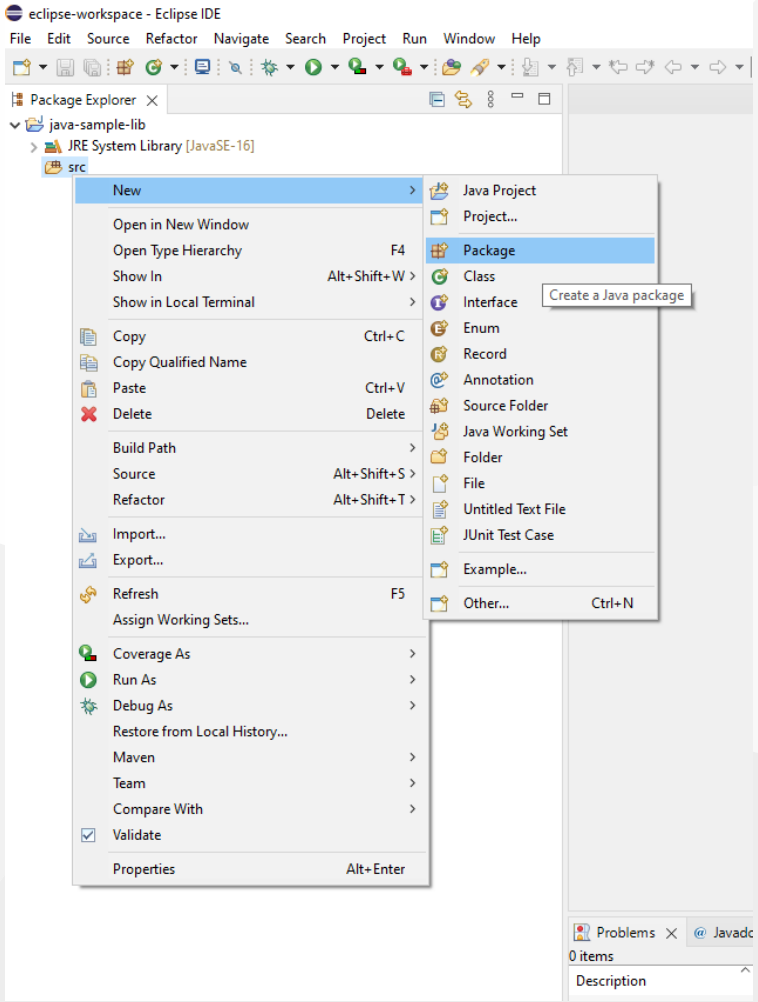
Shared Library Development - (Eclipse Java Jar Library)-8

- select finish



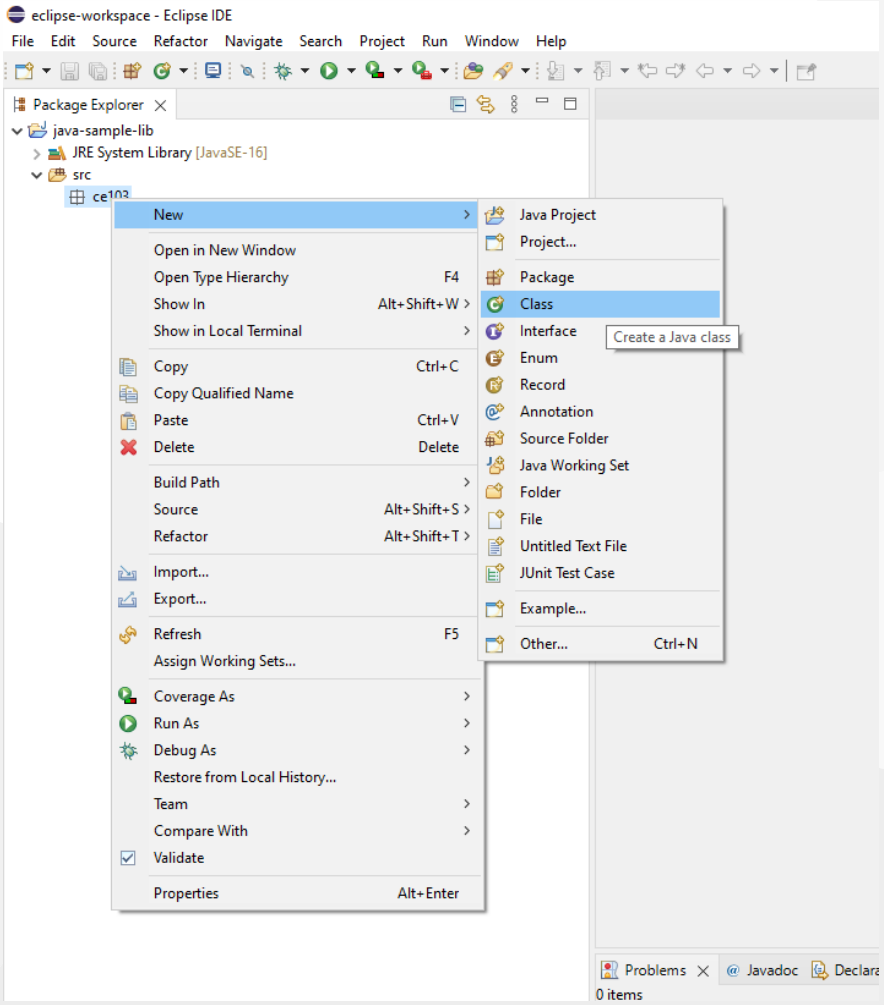
Shared Library Development - (Eclipse Java Jar Library)-9

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



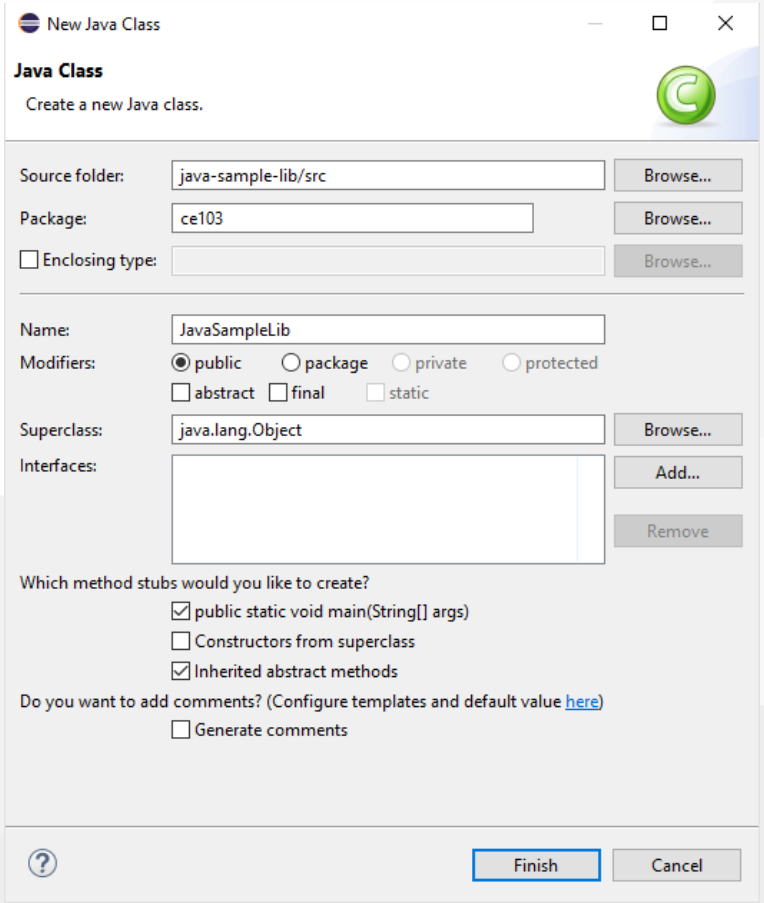
Shared Library Development - (Eclipse Java Jar Library)-10

- then we can create our class that includes our functions



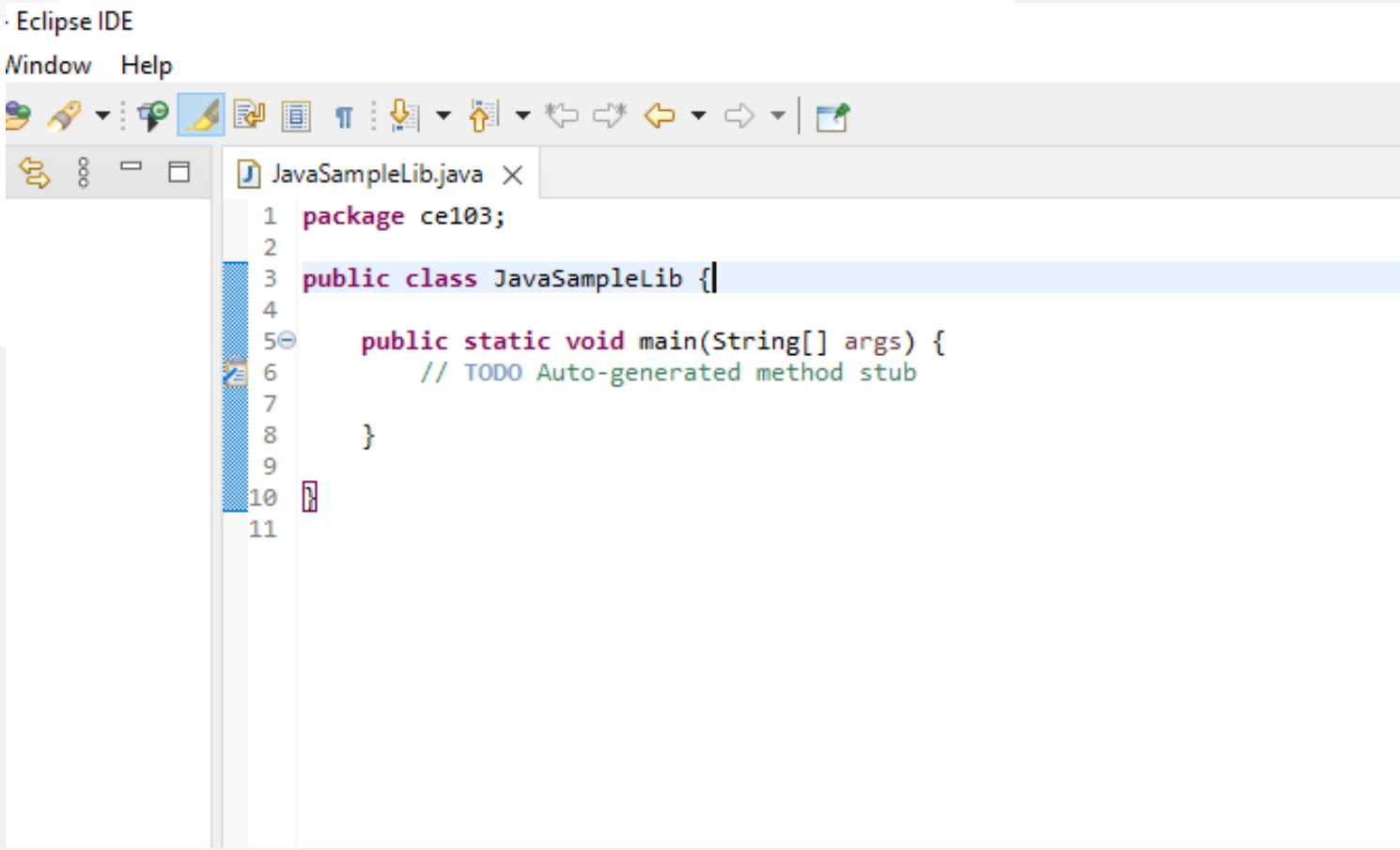
Shared Library Development - (Eclipse Java Jar Library)-11

- give class a name



Shared Library Development - (Eclipse Java Jar Library)-12

- you will have following class with main



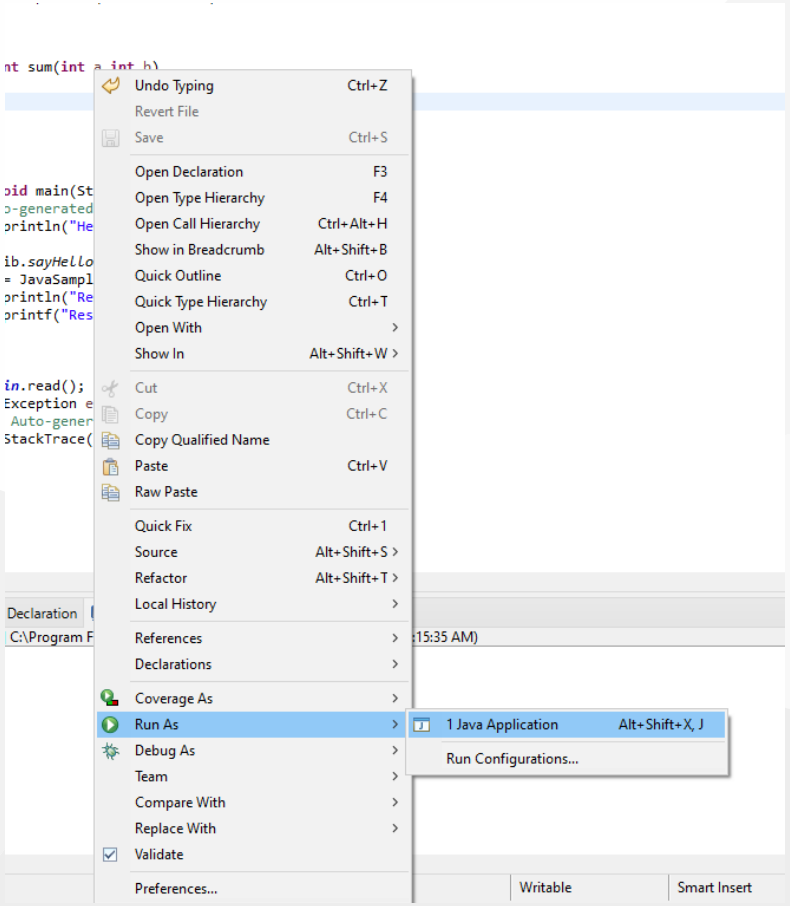
Shared Library Development - (Eclipse Java Jar Library)-13

- 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();
        }

    }

}
```

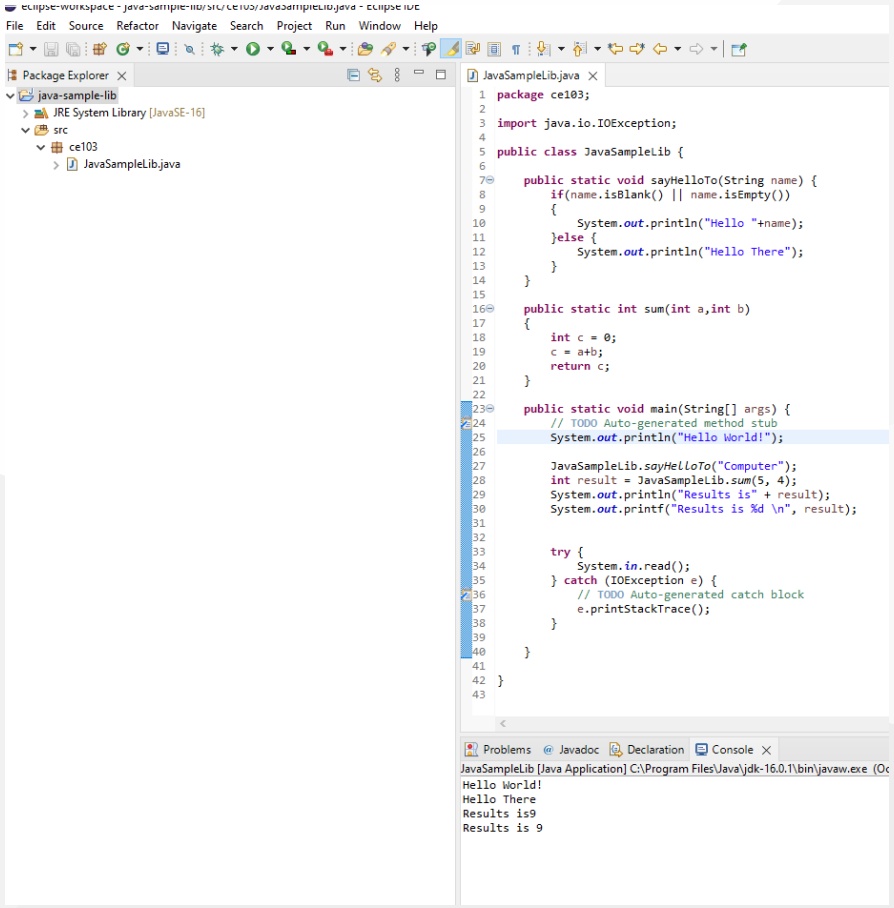
Shared Library Development - (Eclipse Java Jar Library)-14

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



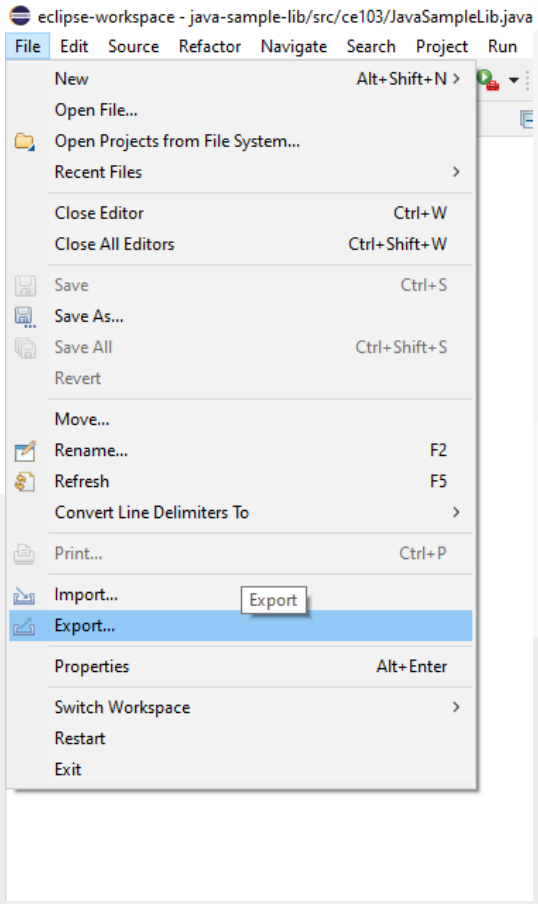
Shared Library Development - (Eclipse Java Jar Library)-15

- we can see output from console as below



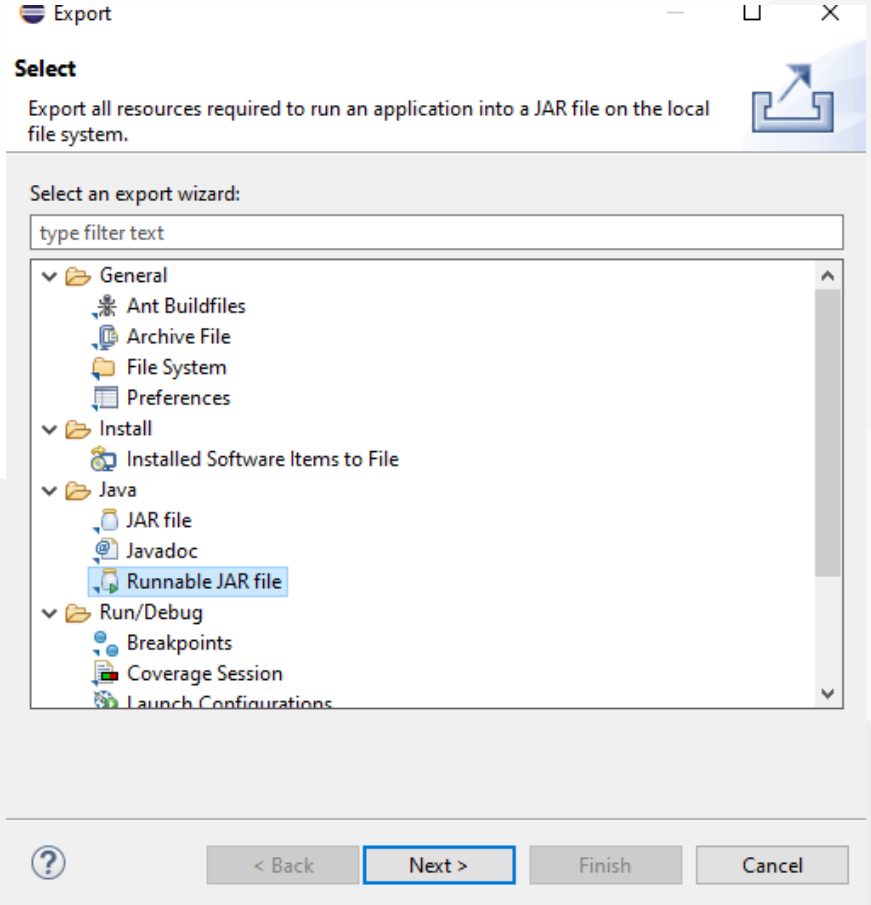
Shared Library Development - (Eclipse Java Jar Library)-16

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



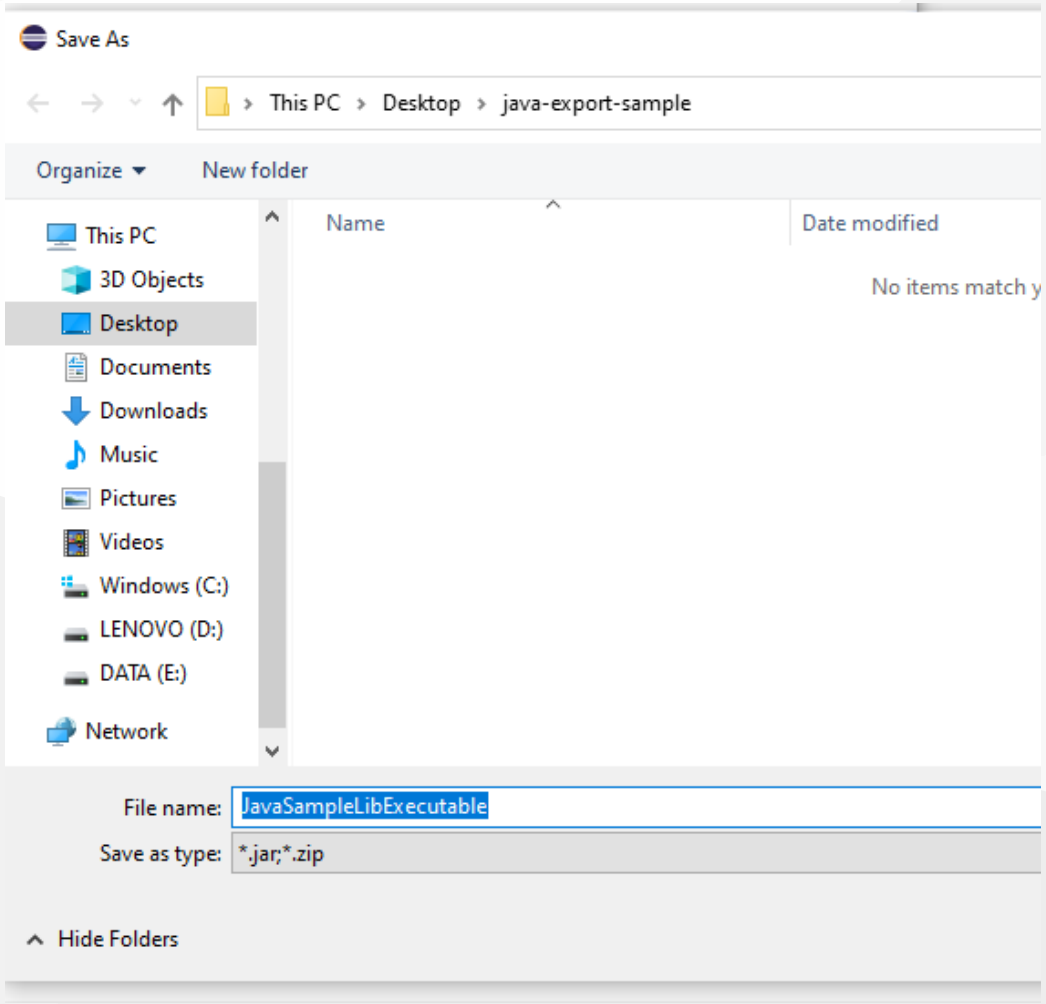
Shared Library Development - (Eclipse Java Jar Library)-17

- Select Java->Runnable JAR File



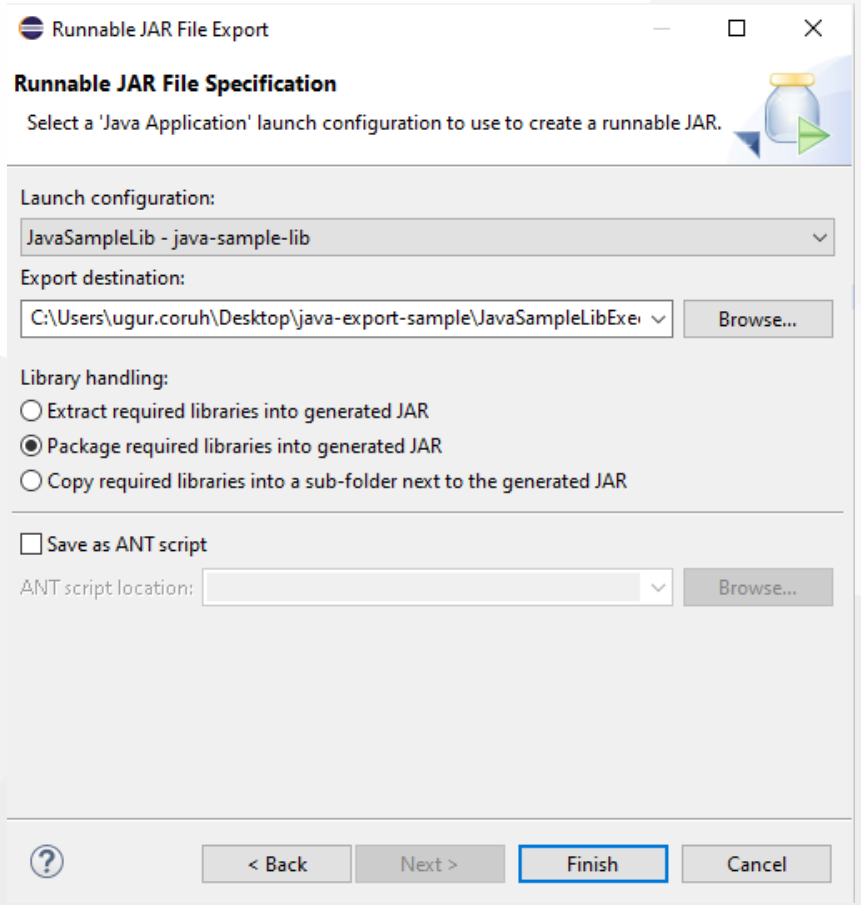
Shared Library Development - (Eclipse Java Jar Library)-18

click next and set output path for jar file



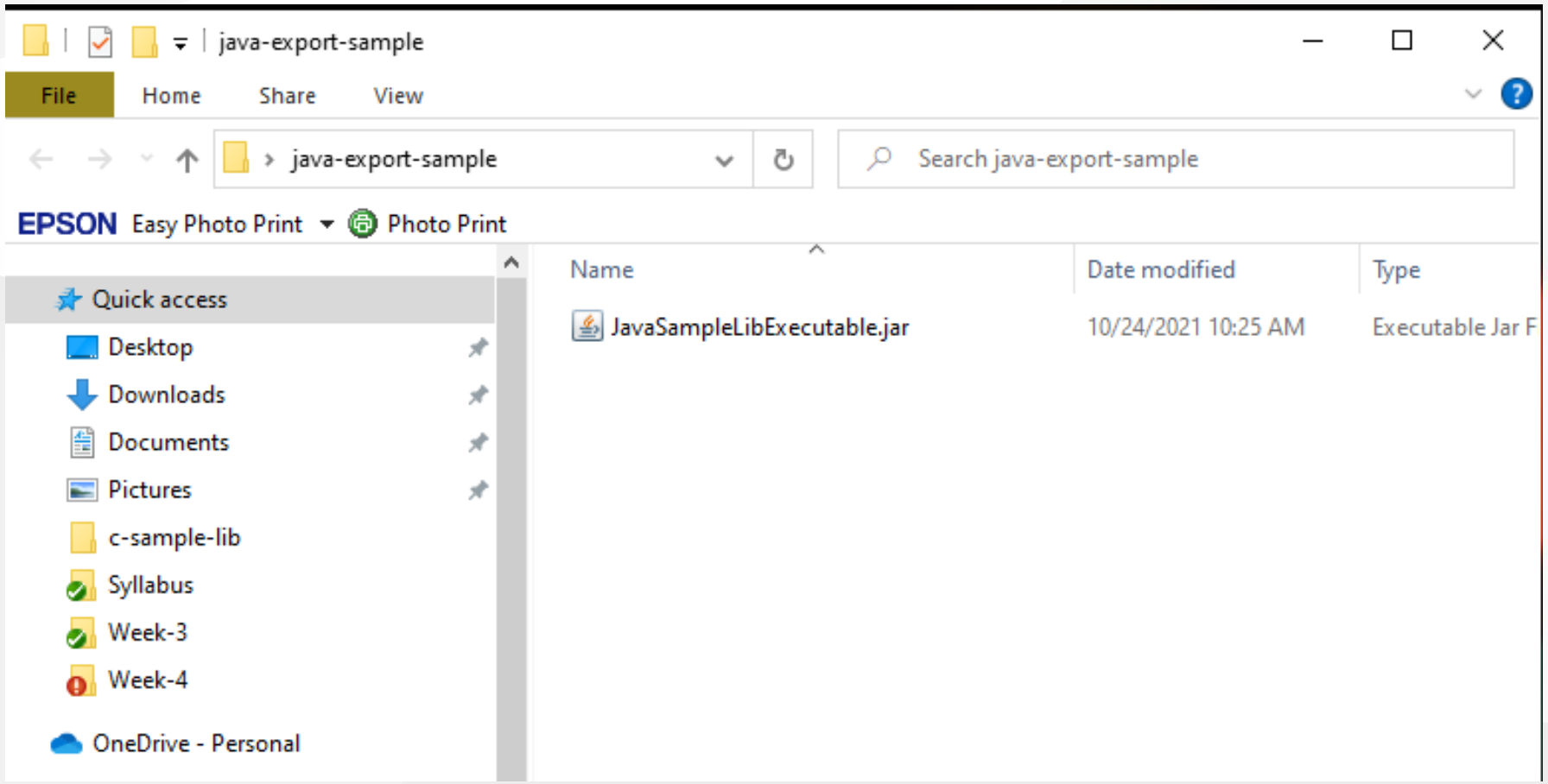
Shared Library Development - (Eclipse Java Jar Library)-19

- 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



Shared Library Development - (Eclipse Java Jar Library)-20

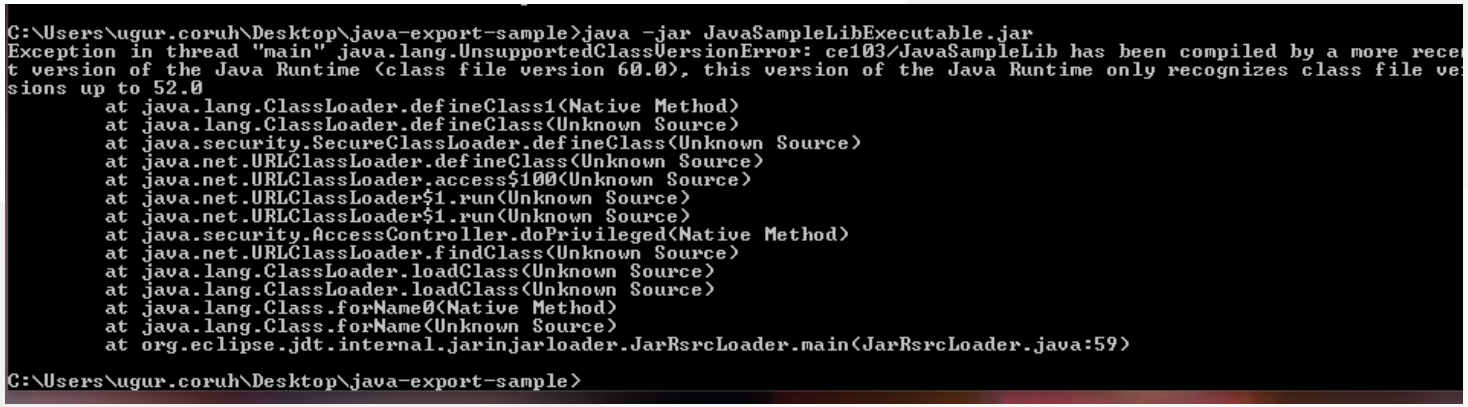
end of this operation we will have the following jar that we can by click



Shared Library Development - (Eclipse Java Jar Library)-21

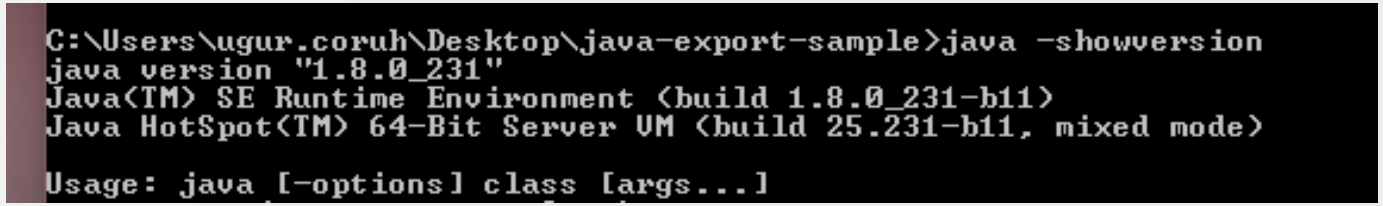
- When you click application if cannot run then try command line to see problem
- enter jar folder and run the following command

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



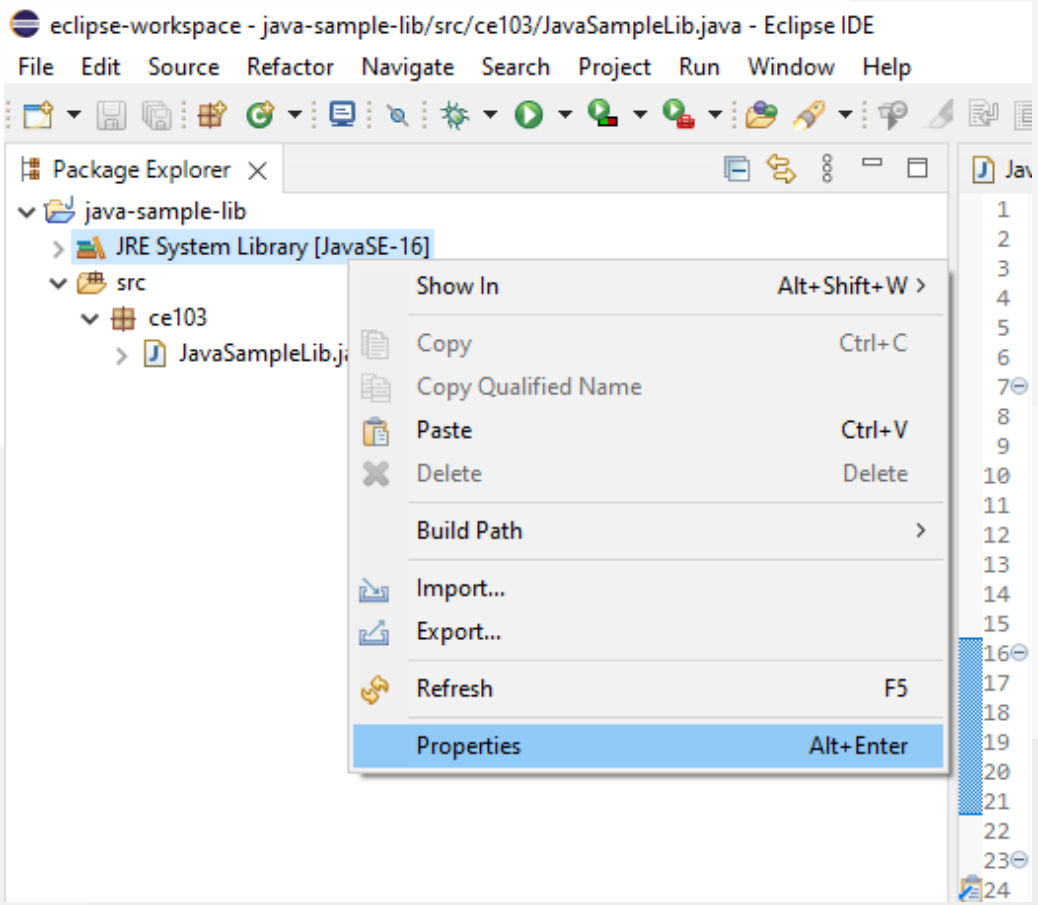
In my case eclipse build JDK is newer than that I installed and set for my OS

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



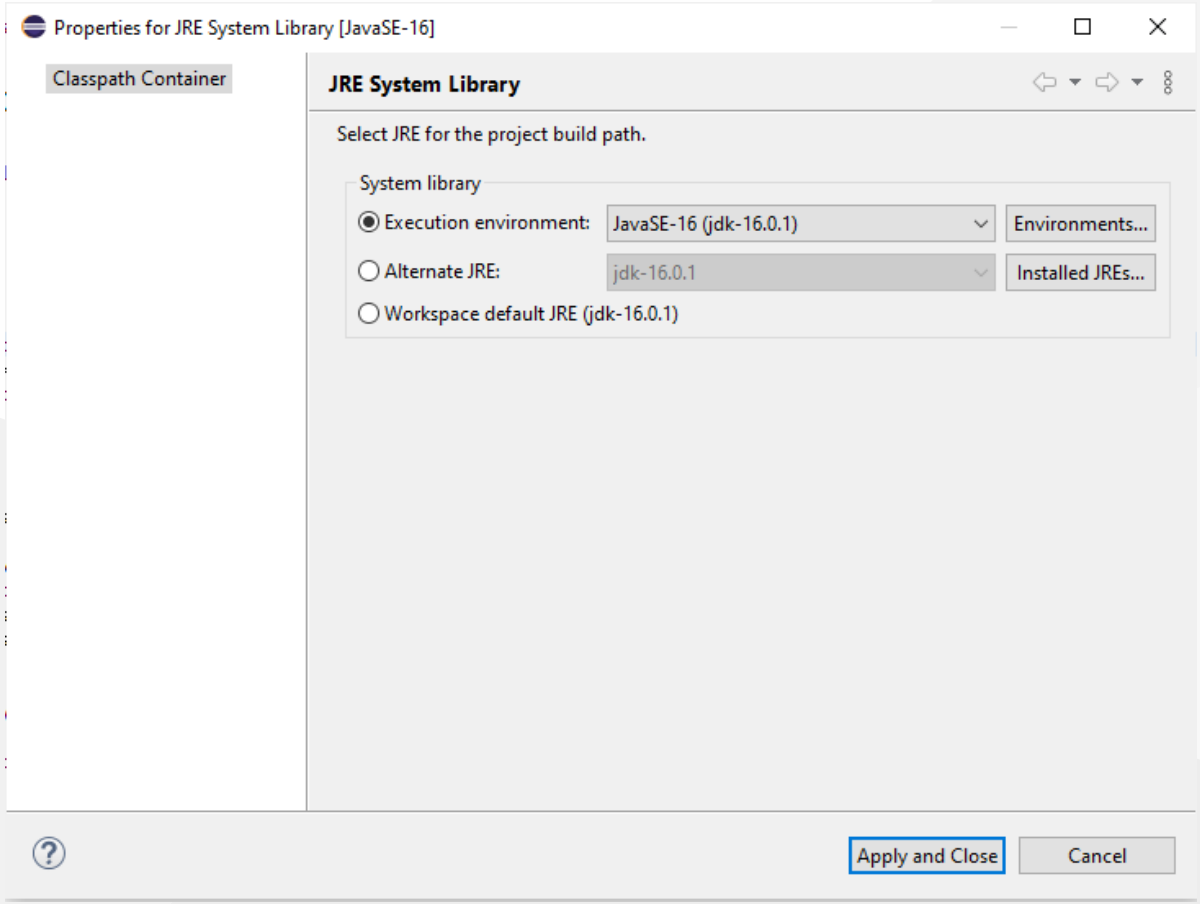
Shared Library Development - (Eclipse Java Jar Library)-22

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



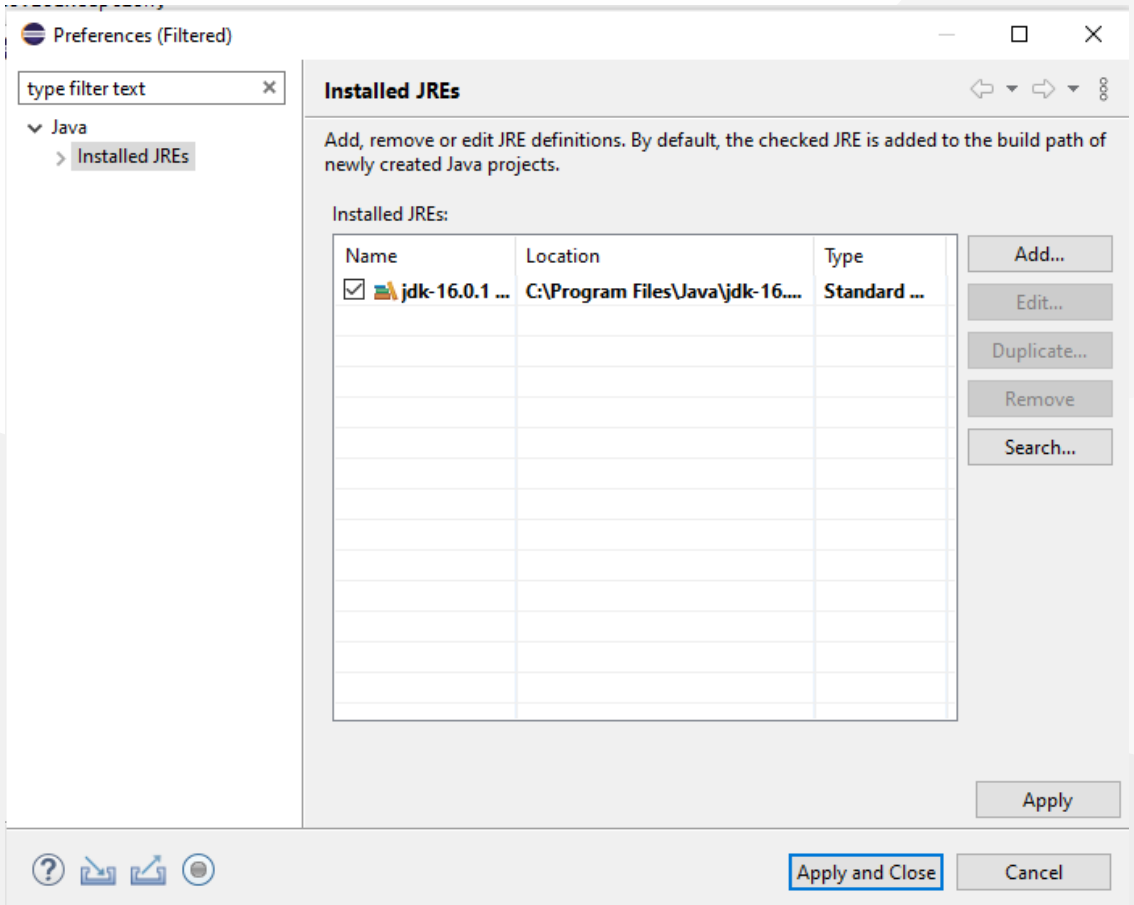
Shared Library Development - (Eclipse Java Jar Library)-23

- select environments



Shared Library Development - (Eclipse Java Jar Library)-24

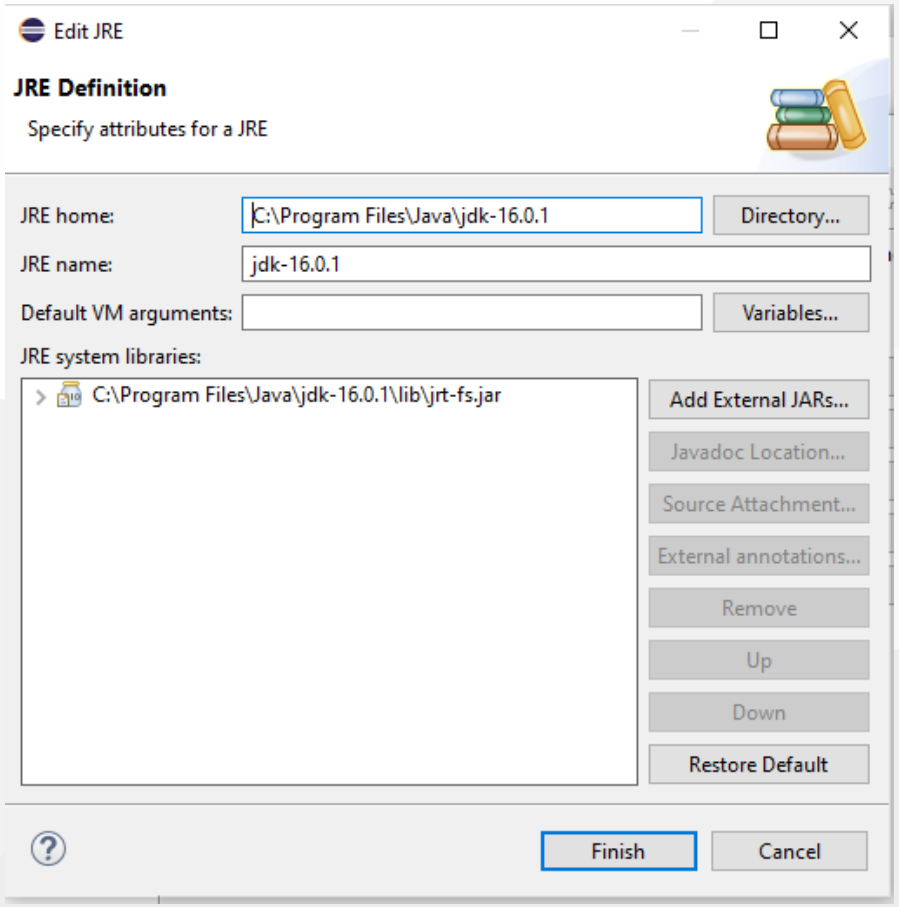
- select installed JRE or JDK



Shared Library Development - (Eclipse Java Jar Library)-25

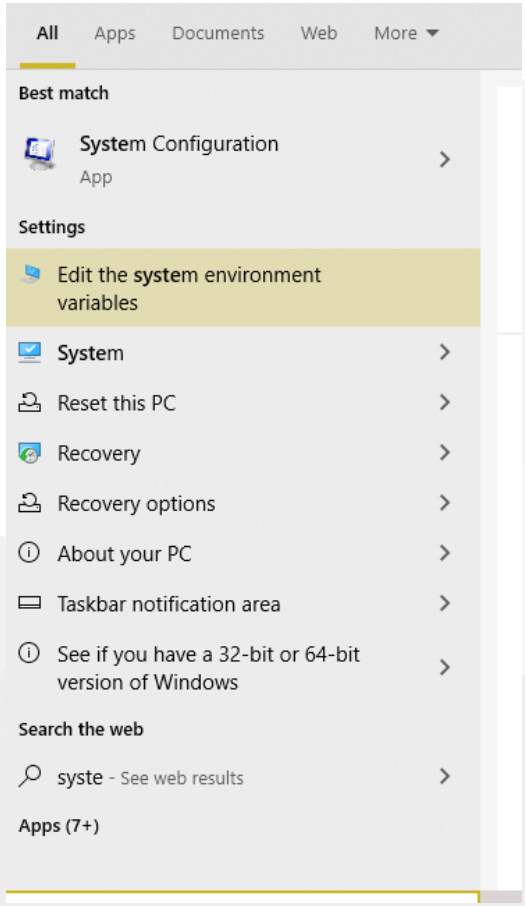
- you can see installed JRE or JDK home

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

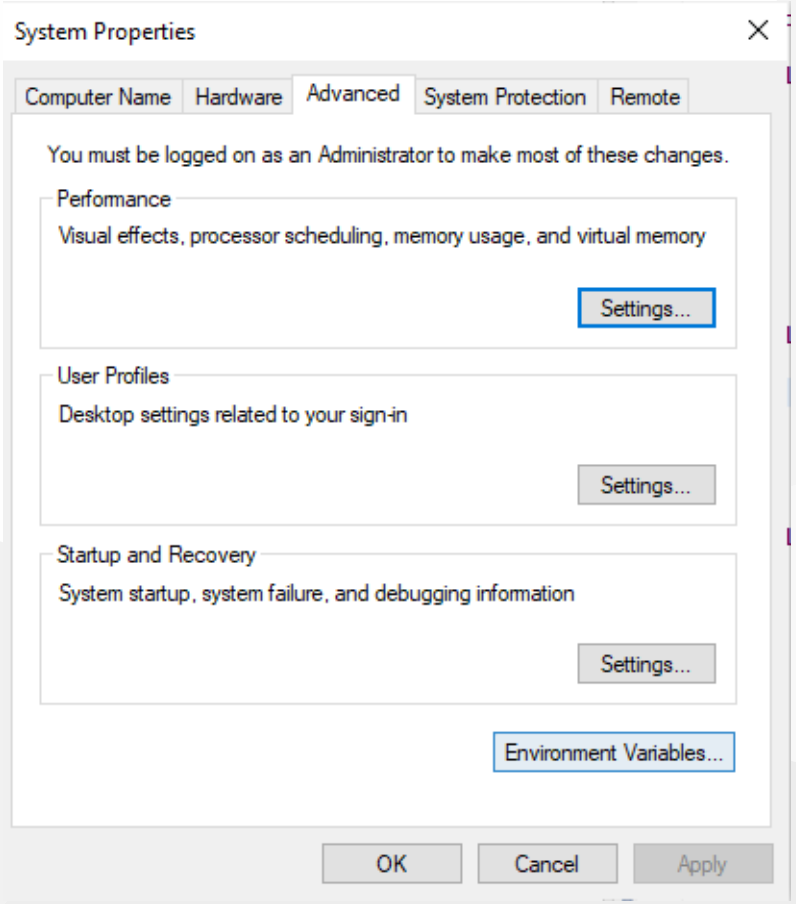


Shared Library Development - (Eclipse Java Jar Library)-26

- Open system environment to fix this problem

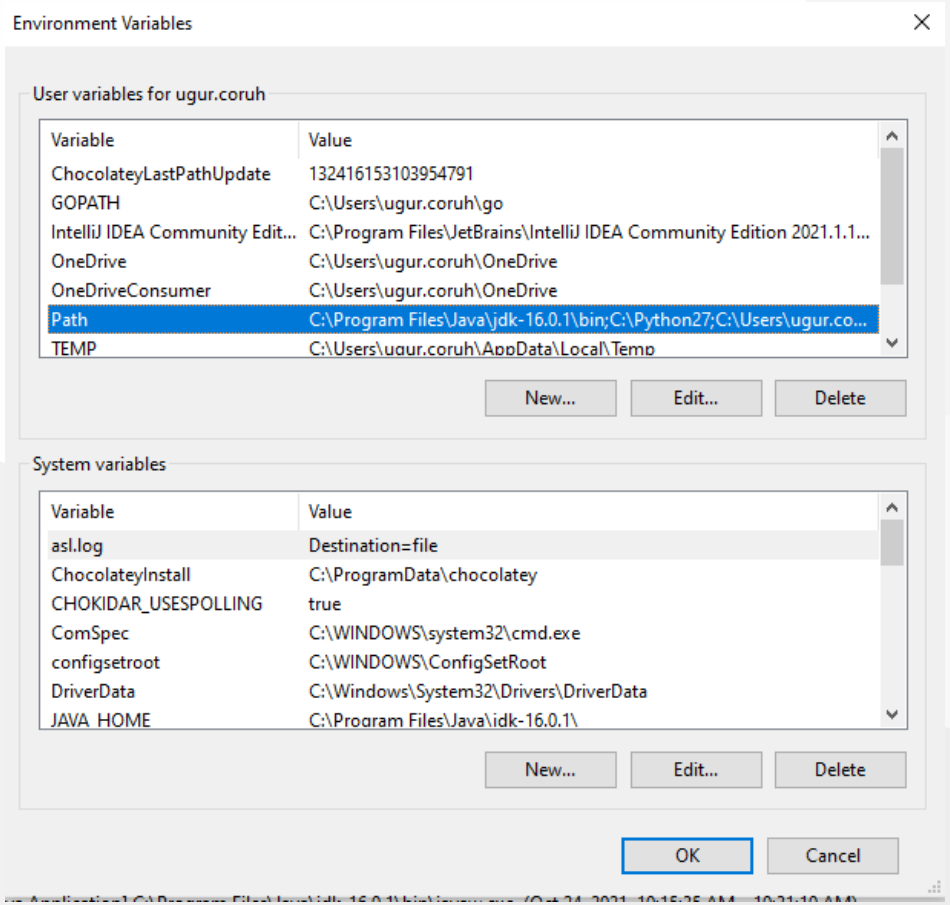


Shared Library Development - (Eclipse Java Jar Library)-27

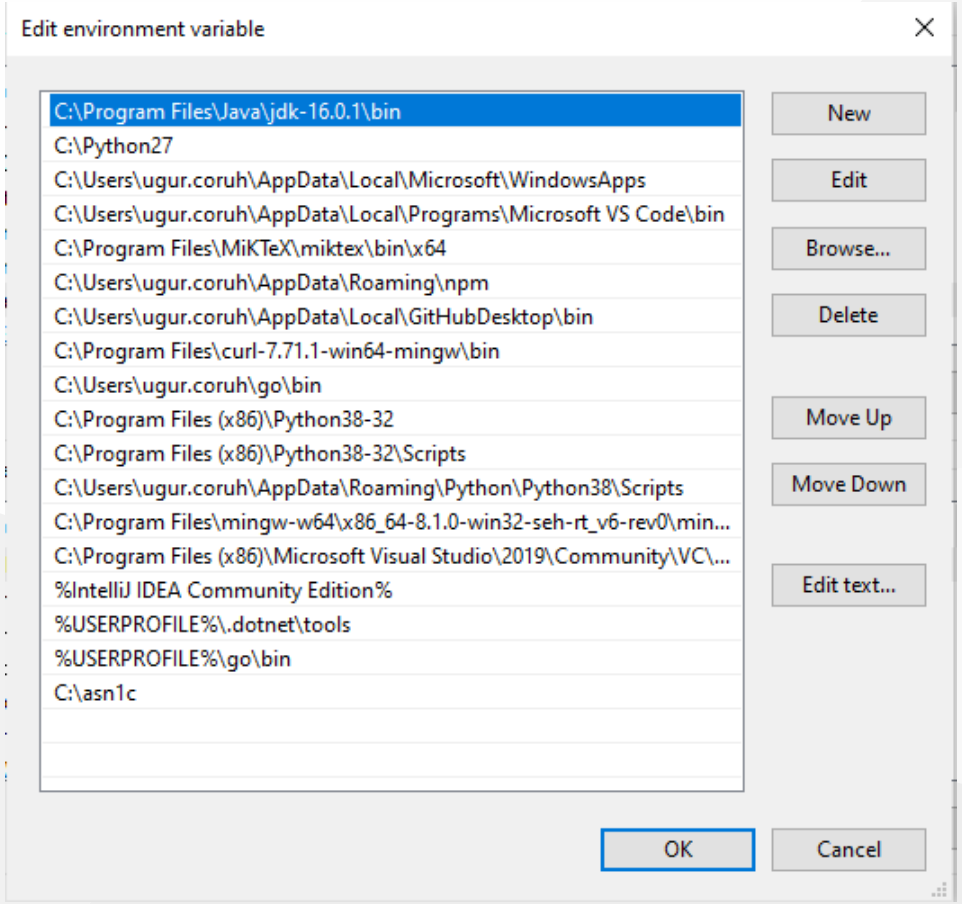


Shared Library Development - (Eclipse Java Jar Library)-28

- Check user settings first

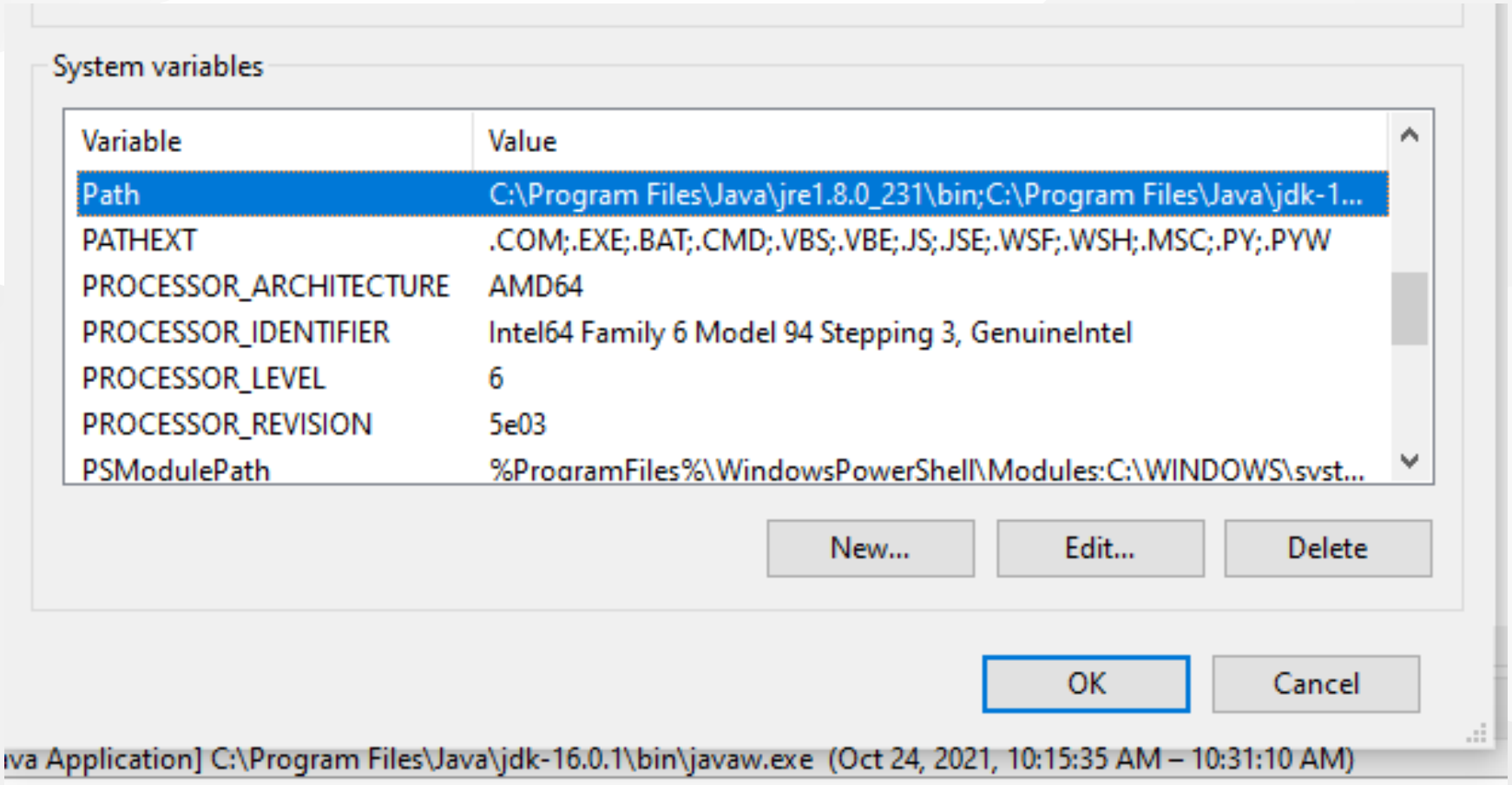


Shared Library Development - (Eclipse Java Jar Library)-29

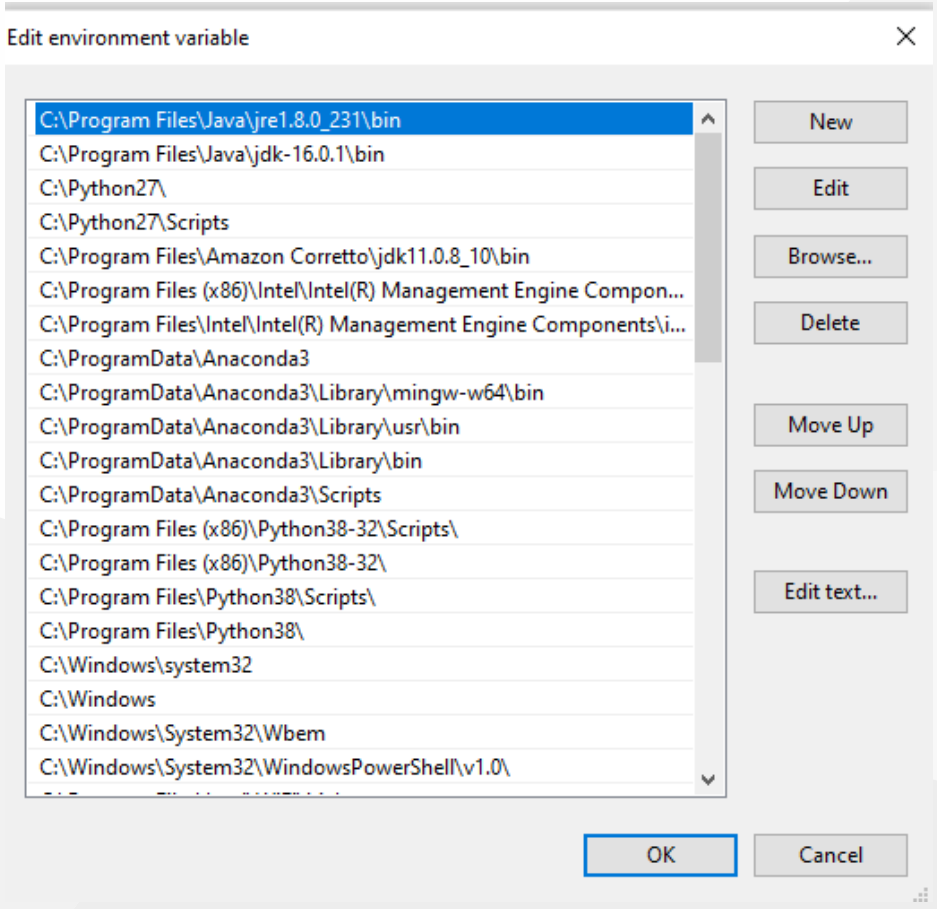


Shared Library Development - (Eclipse Java Jar Library)-30

- Check system settings

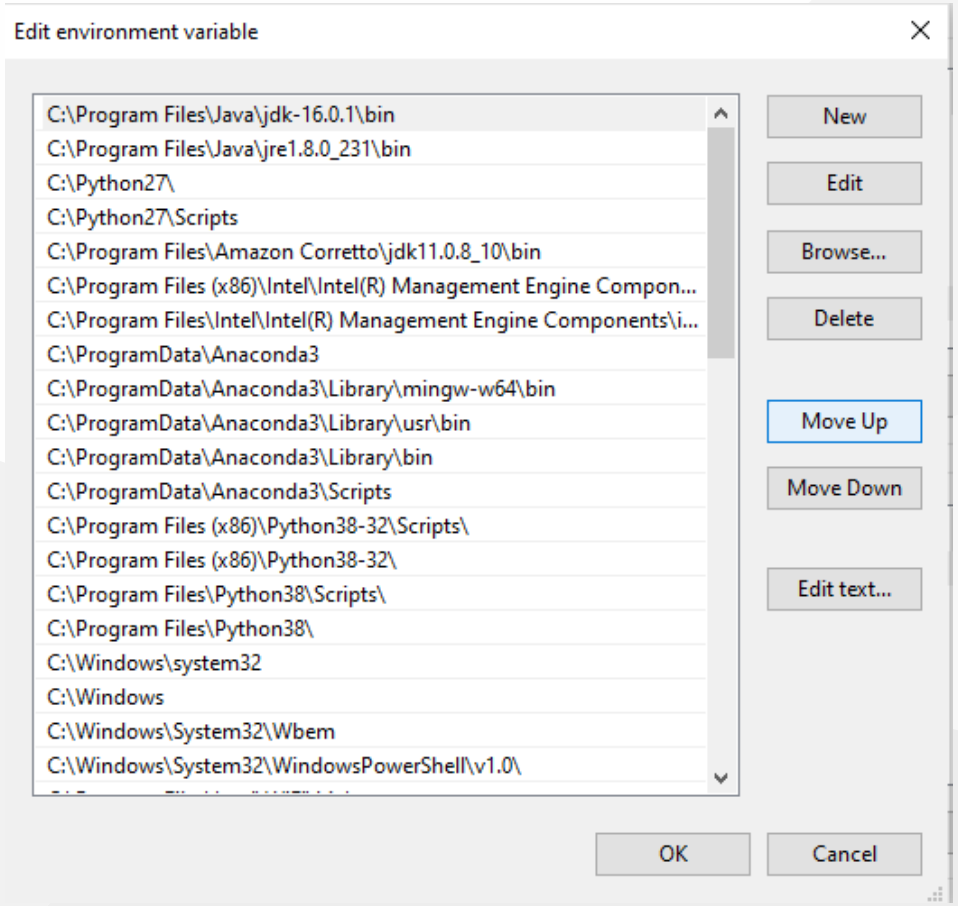


Shared Library Development - (Eclipse Java Jar Library)-31



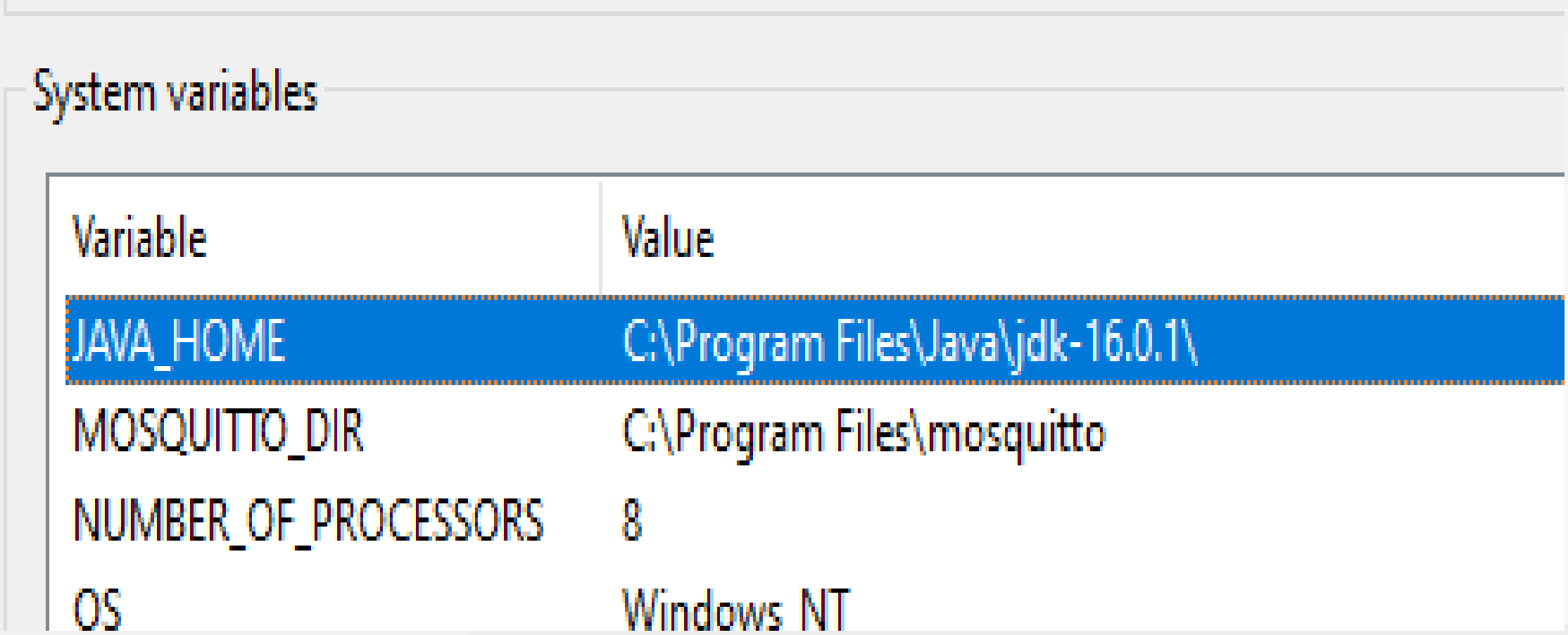
Shared Library Development - (Eclipse Java Jar Library)-32

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



Shared Library Development - (Eclipse Java Jar Library)-33

- Also in system setting check JAVA_HOME

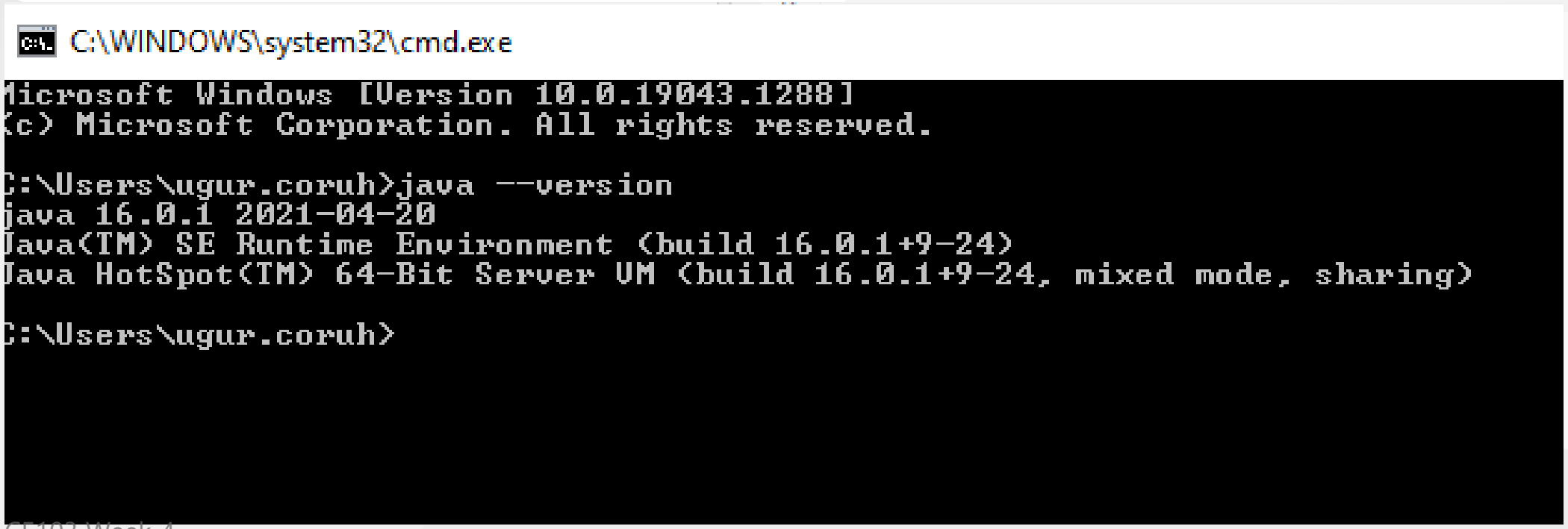


Shared Library Development - (Eclipse Java Jar Library)-34

- 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



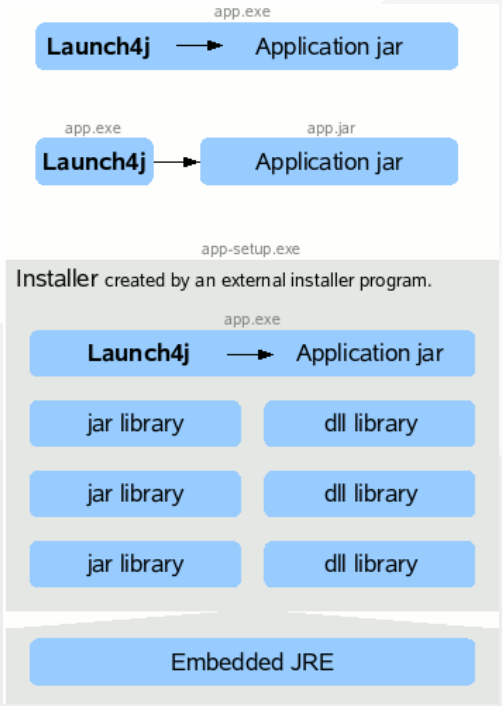
Shared Library Development - (Eclipse Java Jar Library)-35

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

```
C:\Users\ugur.coruh>cd Desktop

C:\Users\ugur.coruh\Desktop>cd java-export-sample

C:\Users\ugur.coruh\Desktop\java-export-sample>java -jar JavaSampleLibExecutable.jar
Hello World!
Hello There
Results is9
Results is 9
```

Shared Library Development - (Eclipse Java Jar Library)-36

- 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

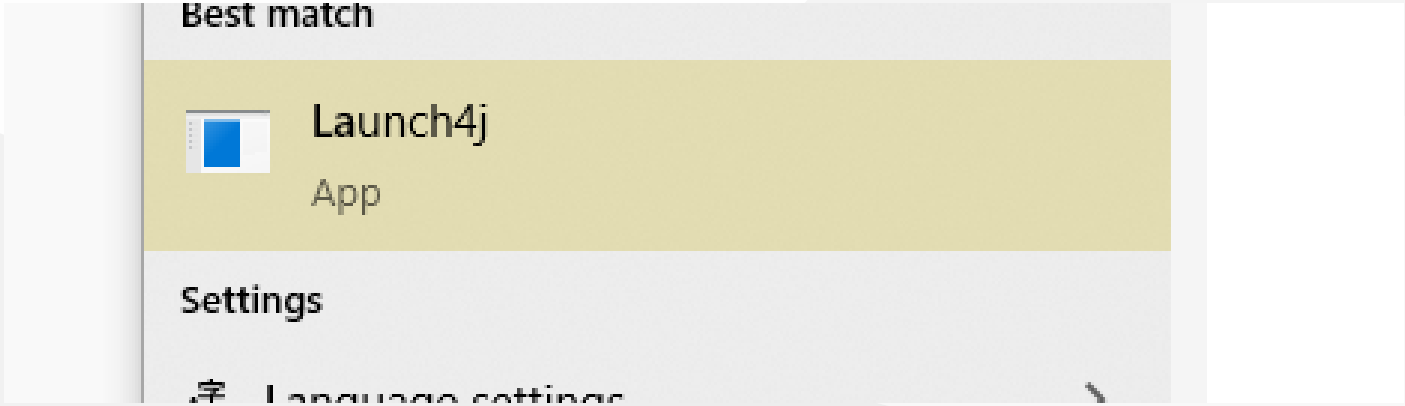


Shared Library Development - (Eclipse Java Jar Library)-37

- you can watch this tutorial also
  - How to convert jar to exe using Launch4J Full explanation - YouTube

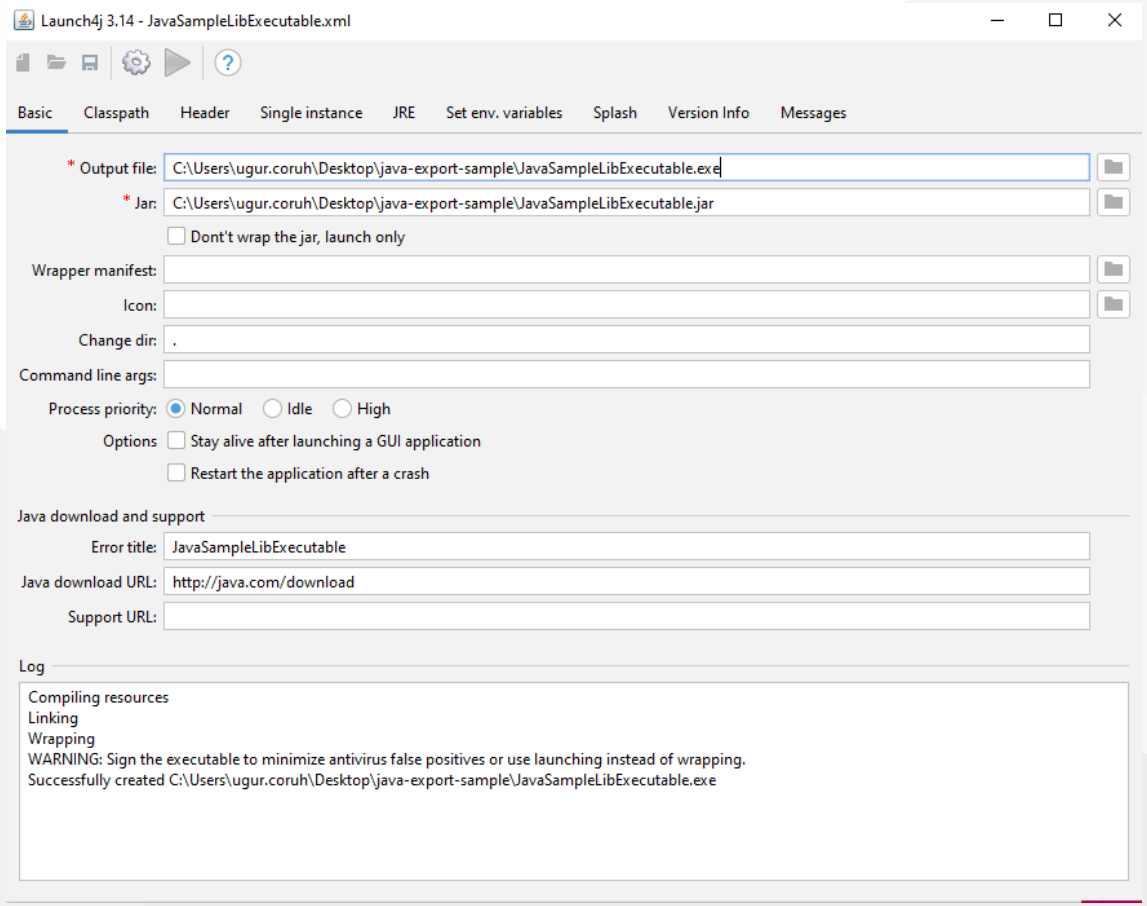
Shared Library Development - (Eclipse Java Jar Library)-38

- Download and install launch4j and open application



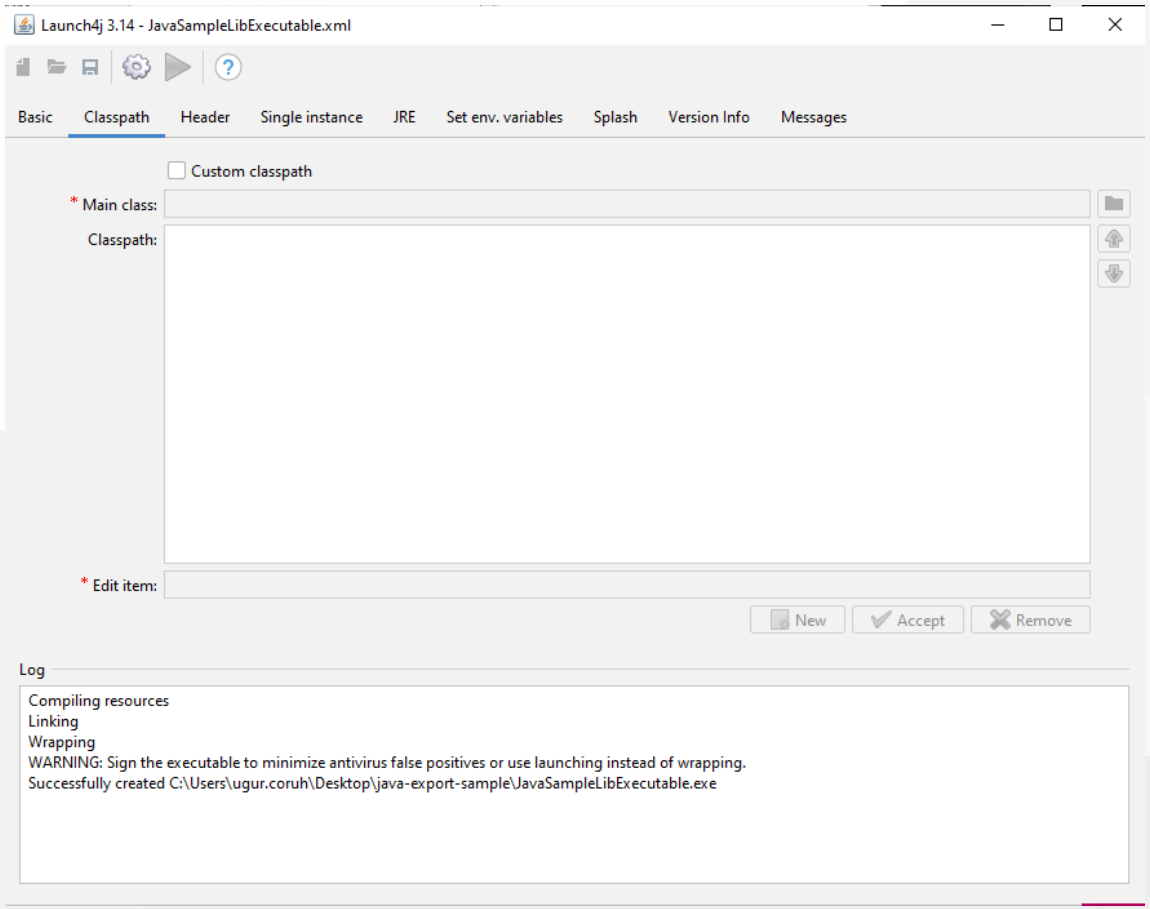
Shared Library Development - (Eclipse Java Jar Library)-39

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



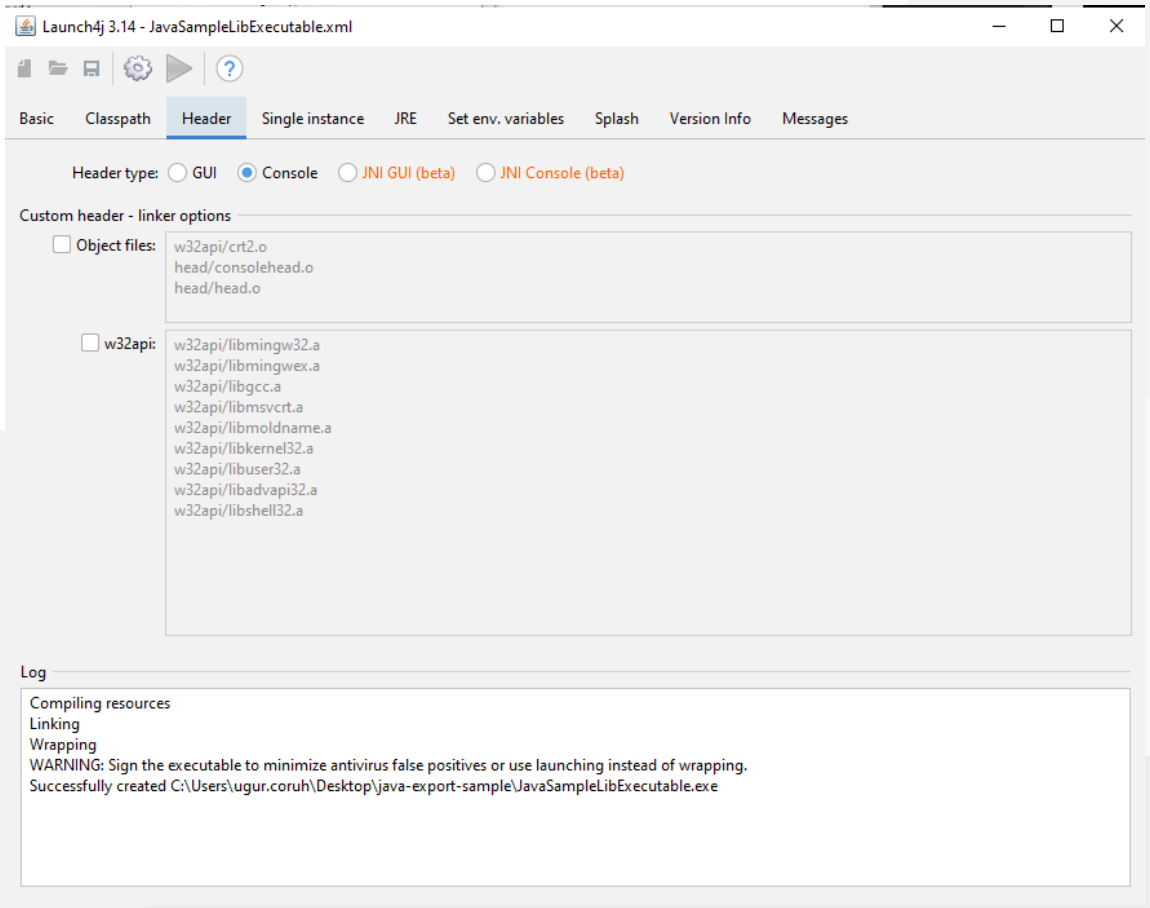
Shared Library Development - (Eclipse Java Jar Library)-40

- We can customize main class if have multiple main class



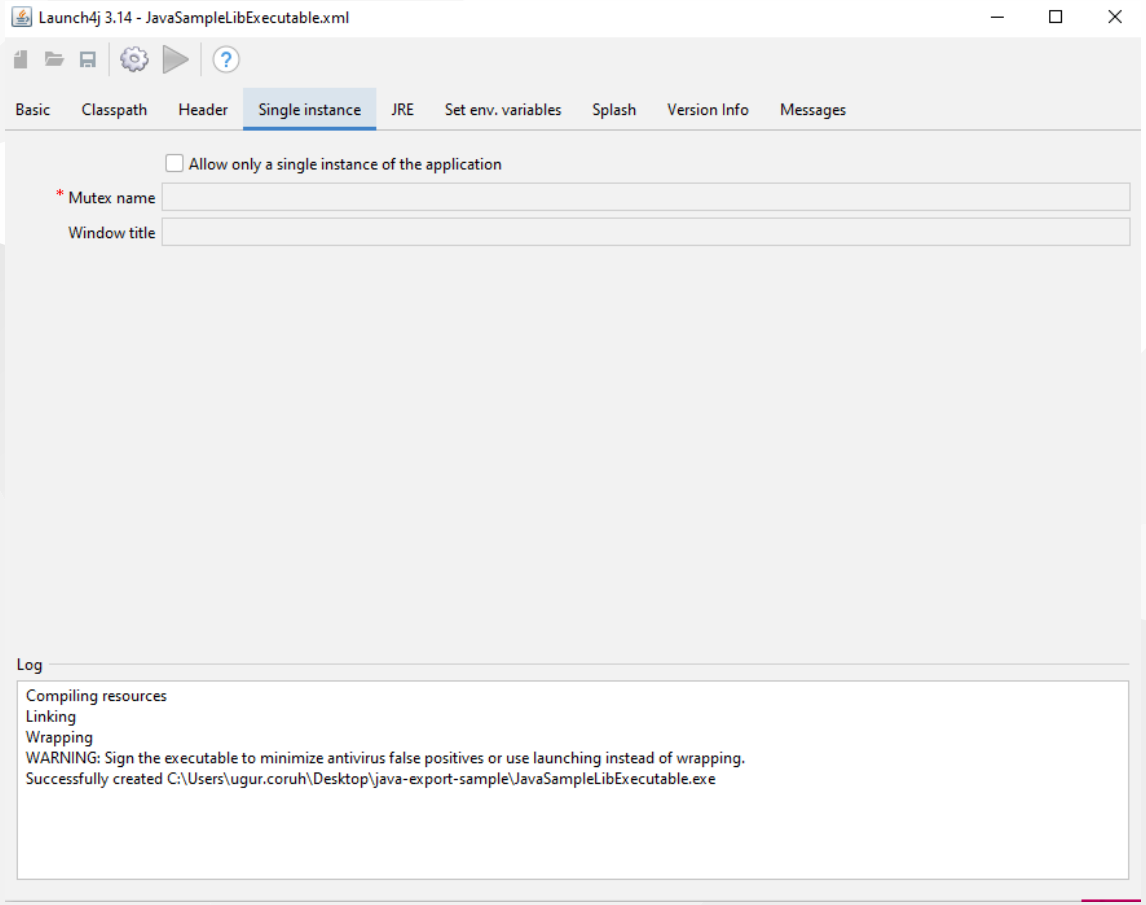
Shared Library Development - (Eclipse Java Jar Library)-41

select console from setting for this application



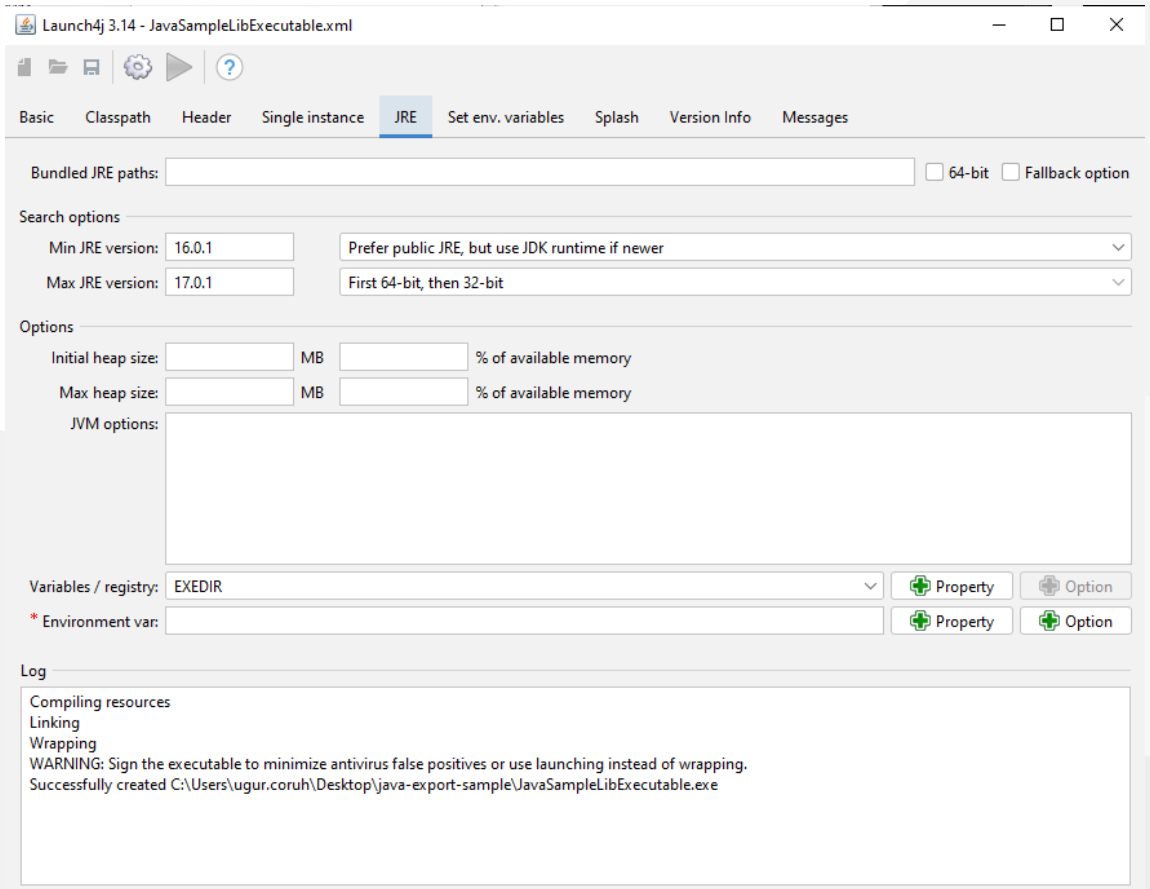
Shared Library Development - (Eclipse Java Jar Library)-42

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



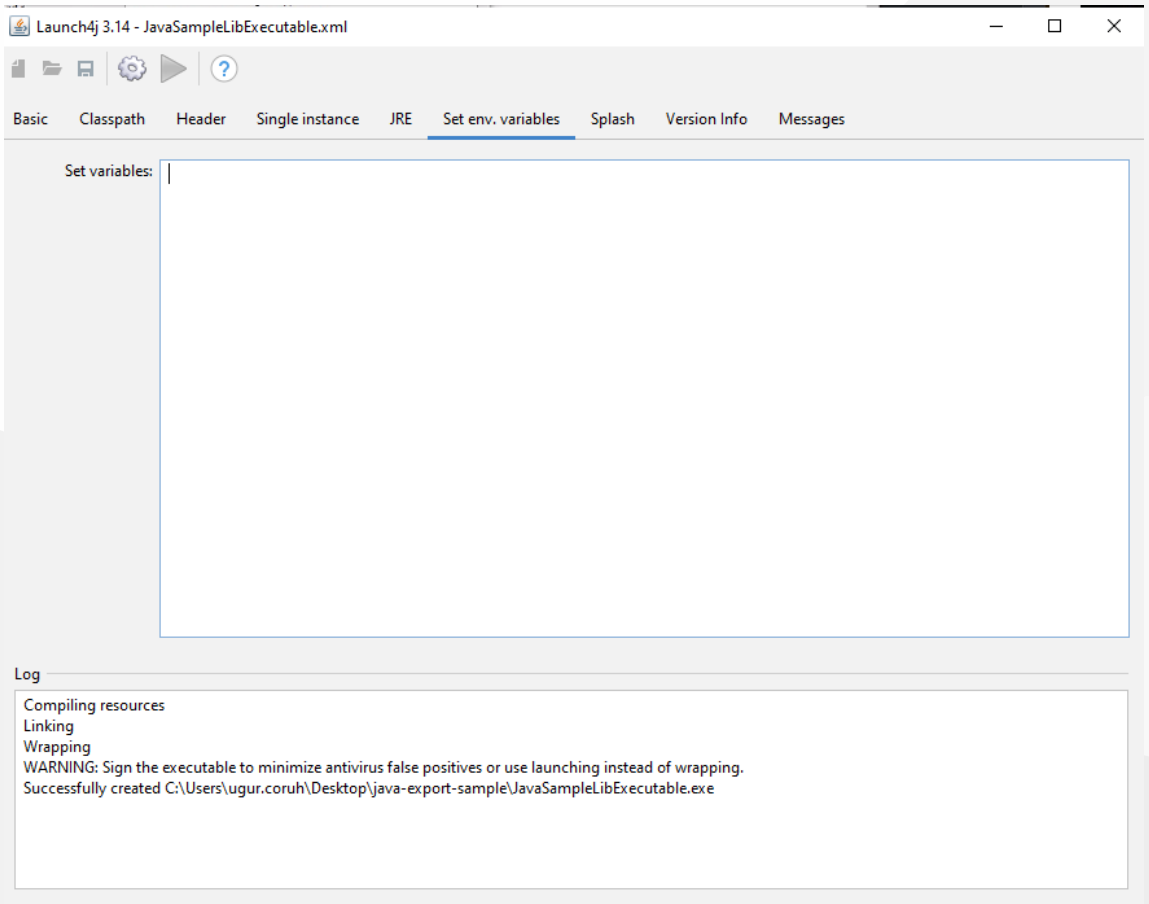
Shared Library Development - (Eclipse Java Jar Library)-43

- we need to set runtime environment versions



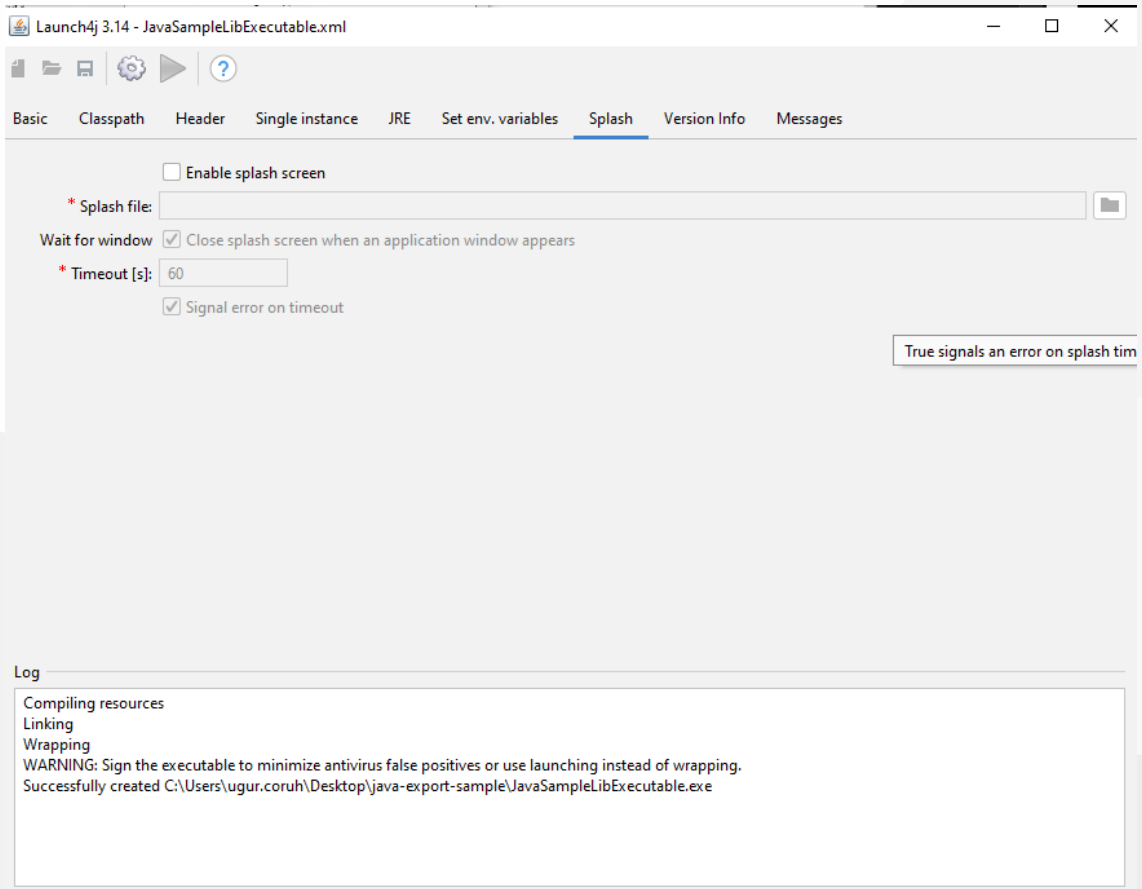
Shared Library Development - (Eclipse Java Jar Library)-44

you can set system parameters before running application



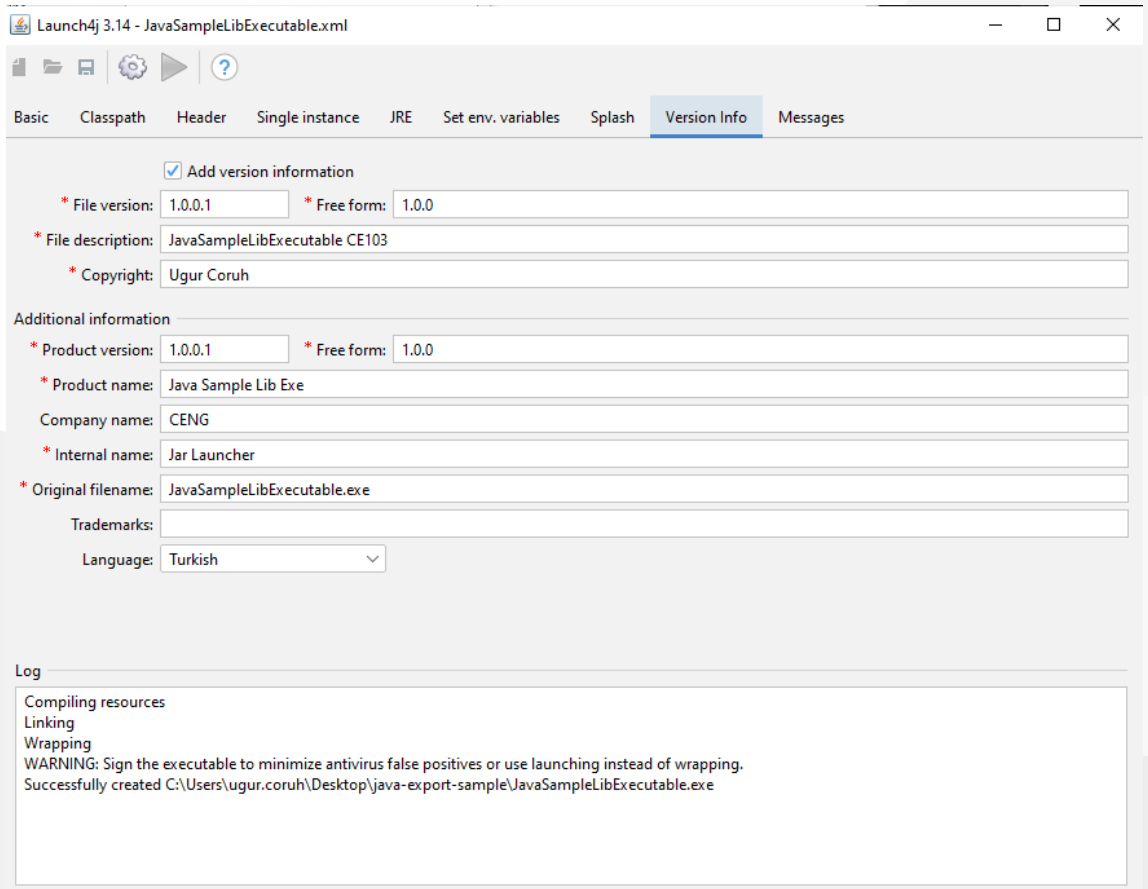
Shared Library Development - (Eclipse Java Jar Library)-45

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



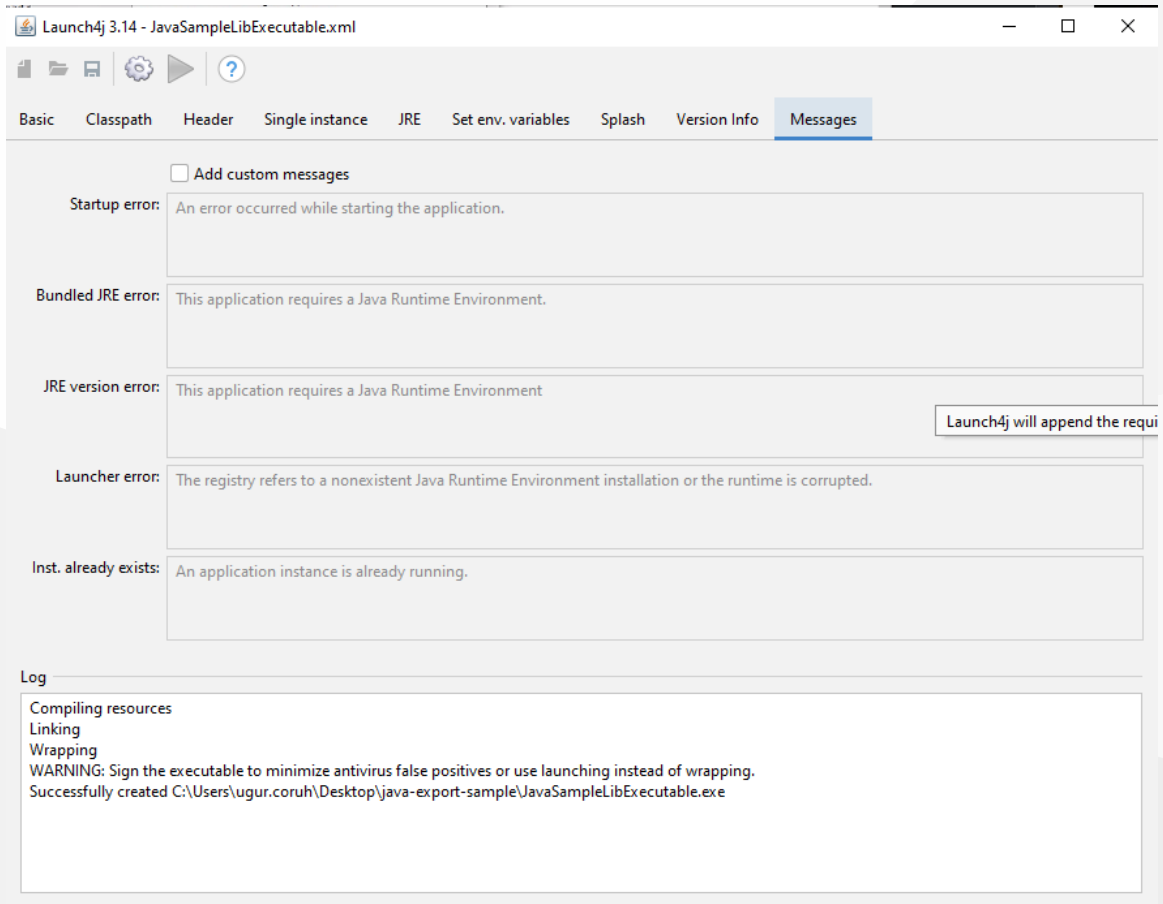
Shared Library Development - (Eclipse Java Jar Library)-46

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



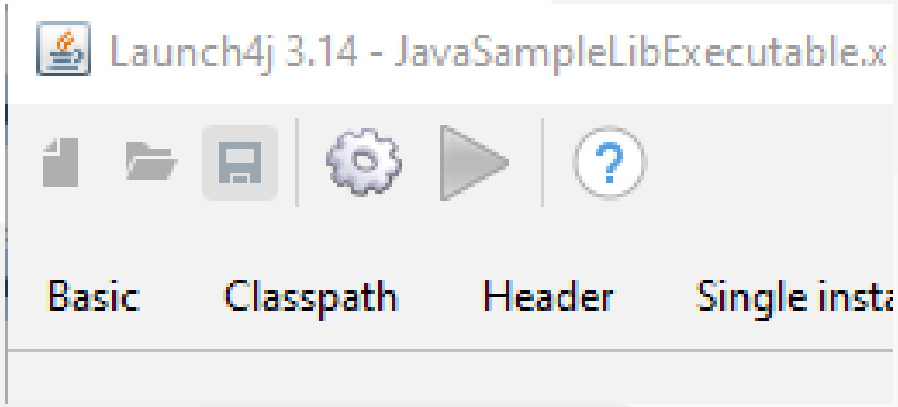
Shared Library Development - (Eclipse Java Jar Library)-47

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



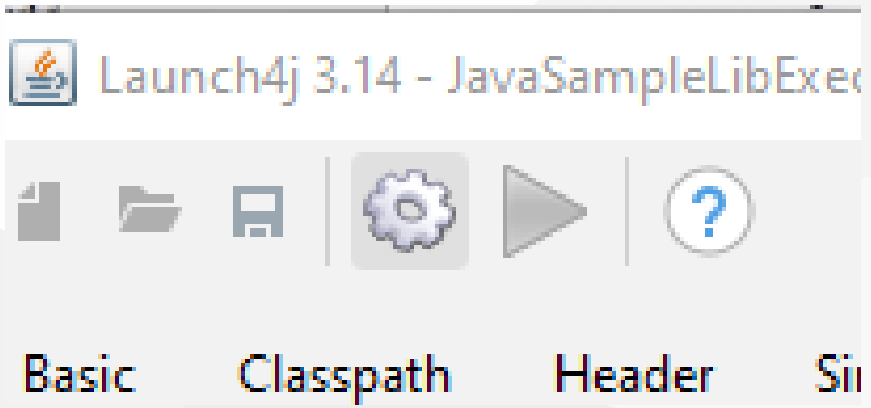
Shared Library Development - (Eclipse Java Jar Library)-48

- with this options save configuration file xml



Shared Library Development - (Eclipse Java Jar Library)-49

- and compile settings



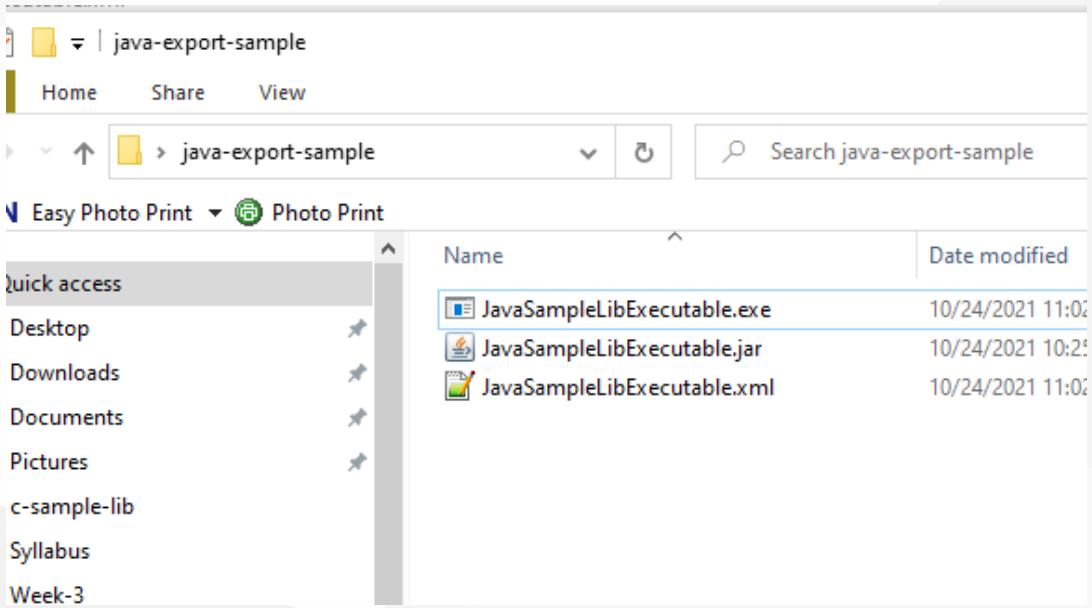
Shared Library Development - (Eclipse Java Jar Library)-50

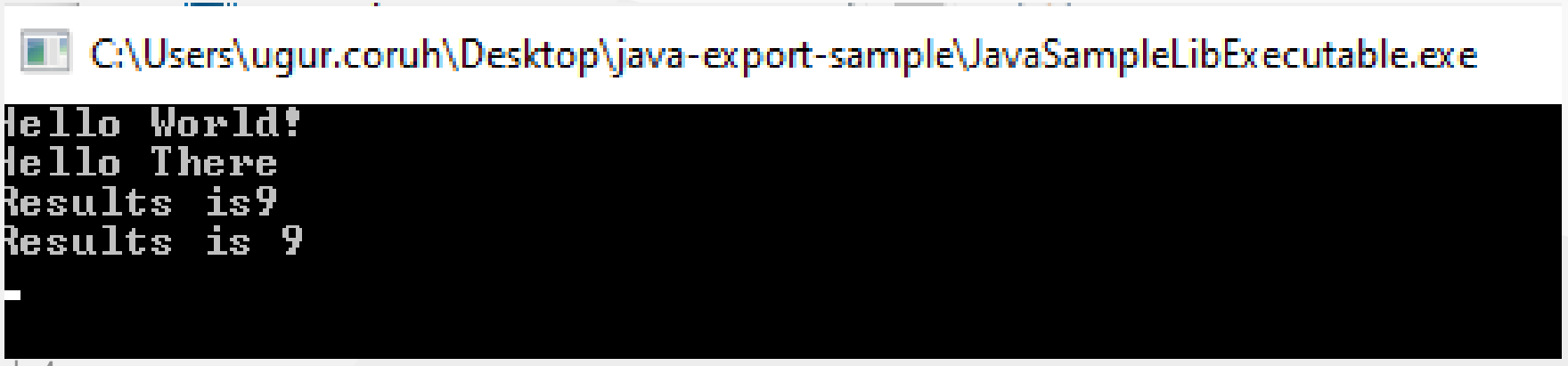
- You will see generated output file in log screen

```
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
```

Shared Library Development - (Eclipse Java Jar Library)-51

- now we can run exe by click





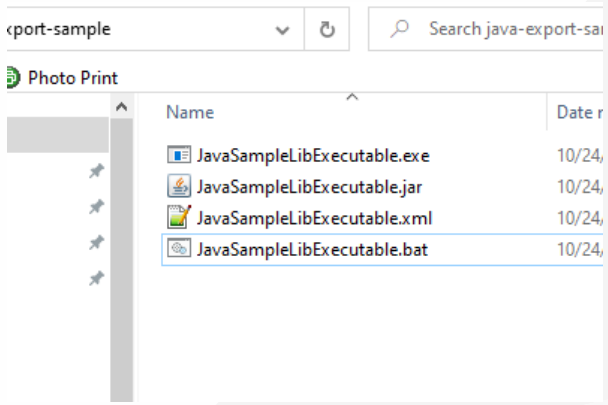
Shared Library Development - (Eclipse Java Jar Library)-52

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

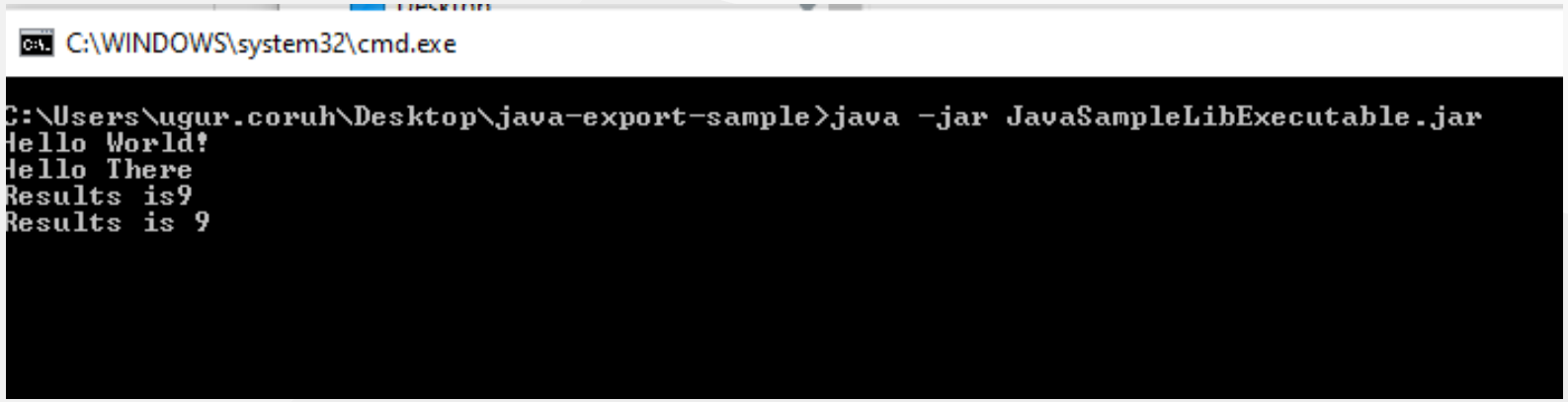
Shared Library Development - (Eclipse Java Jar Library)-53

JavaSampleLibExecutable.bat

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

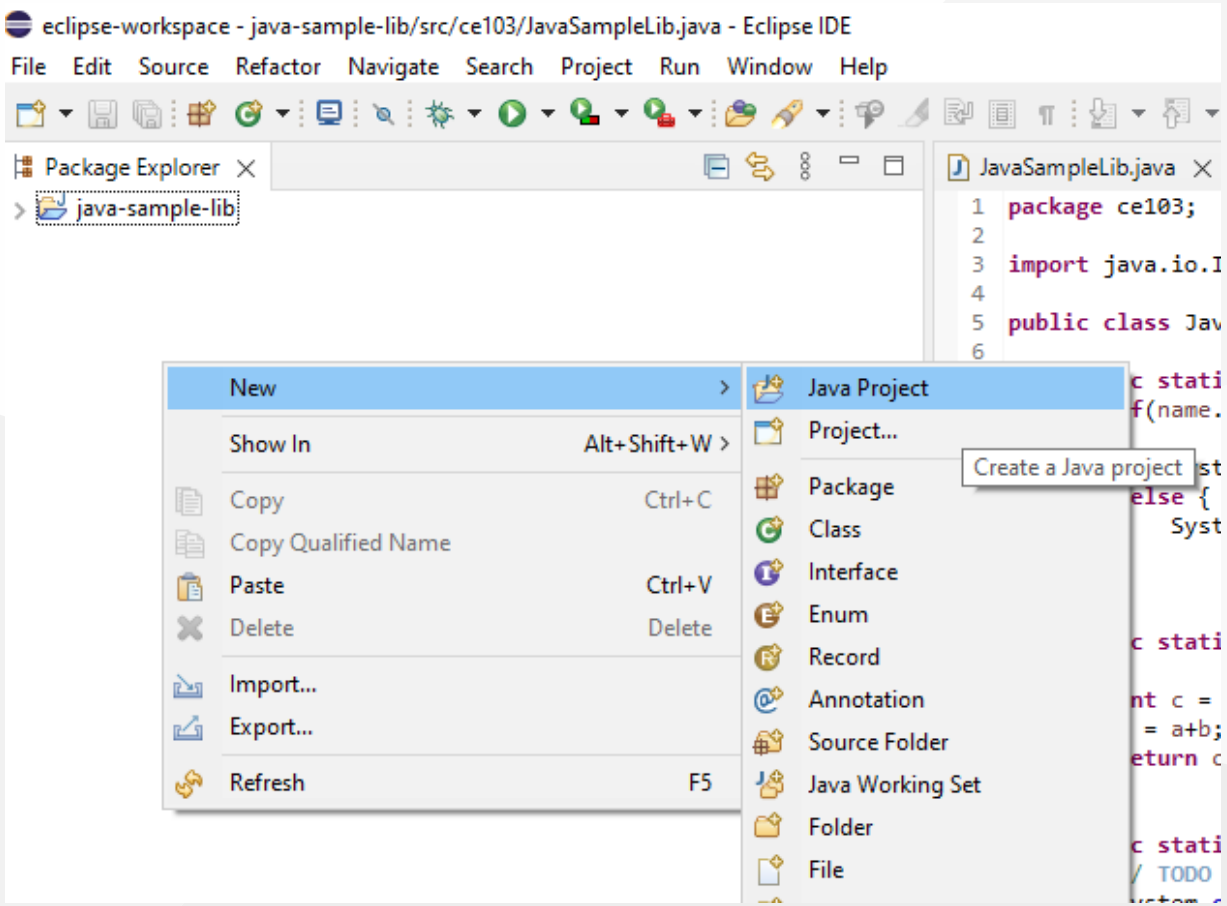


- if we click bat file then we will automate command line task for current jar file

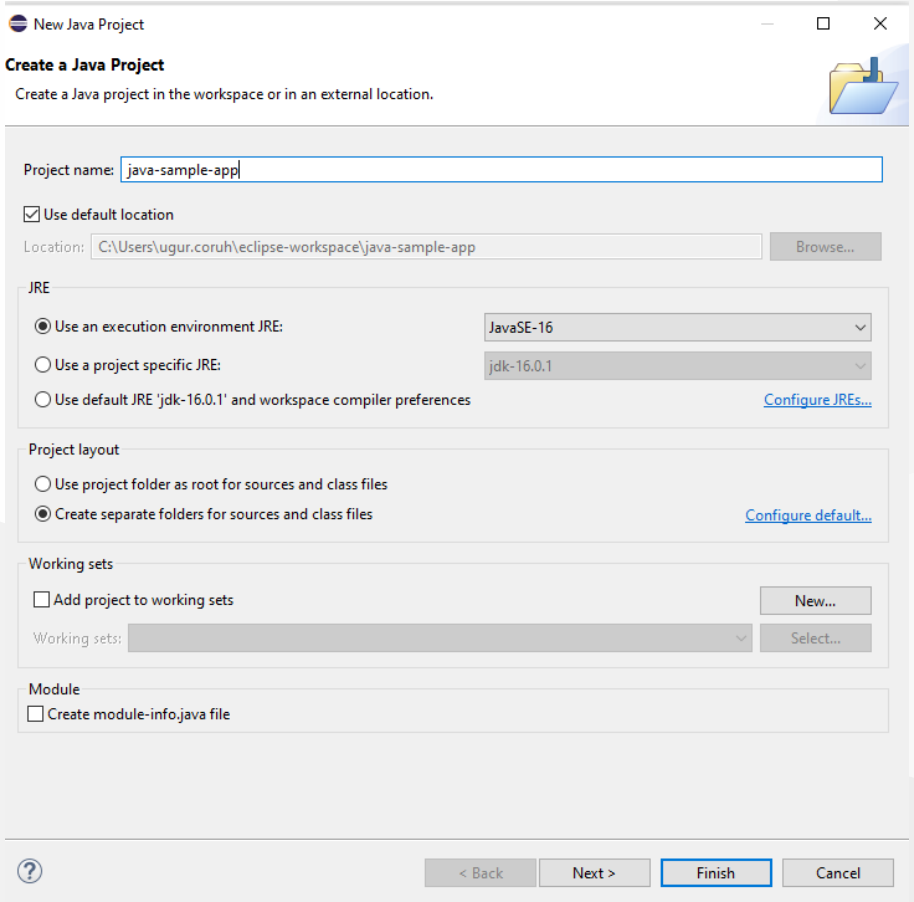


Shared Library Development - (Eclipse Java Jar Library)-54

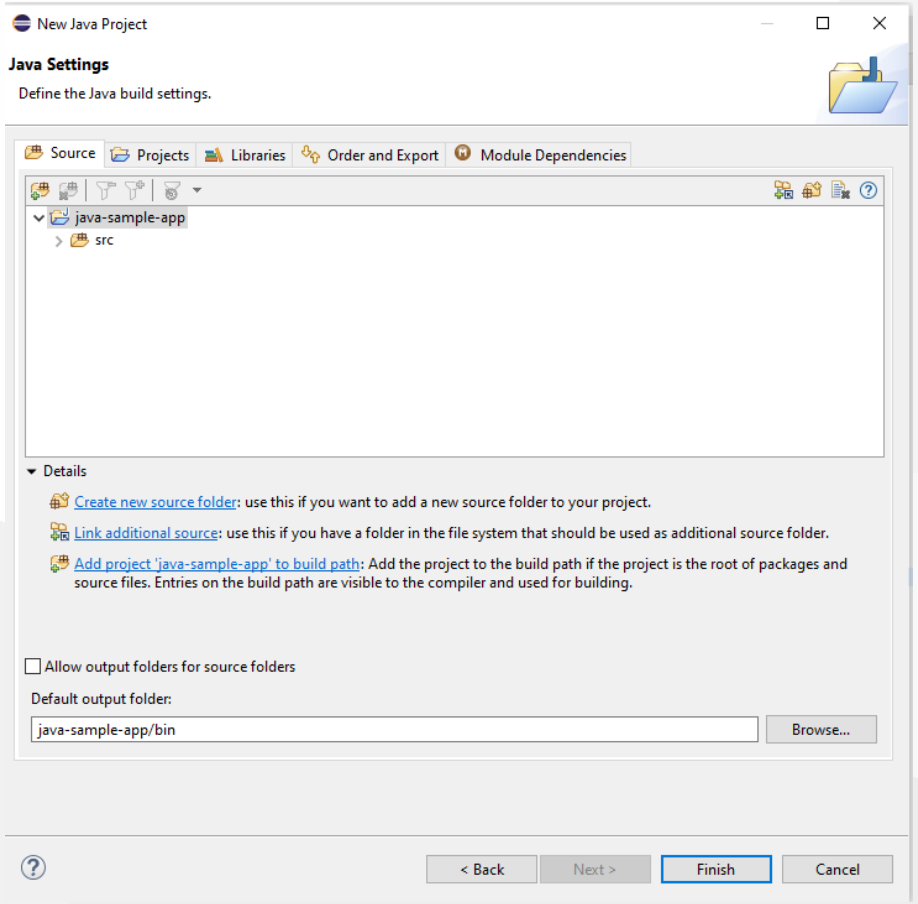
Now return back to our java library and create another console application that use library functions



Shared Library Development - (Eclipse Java Jar Library)-55

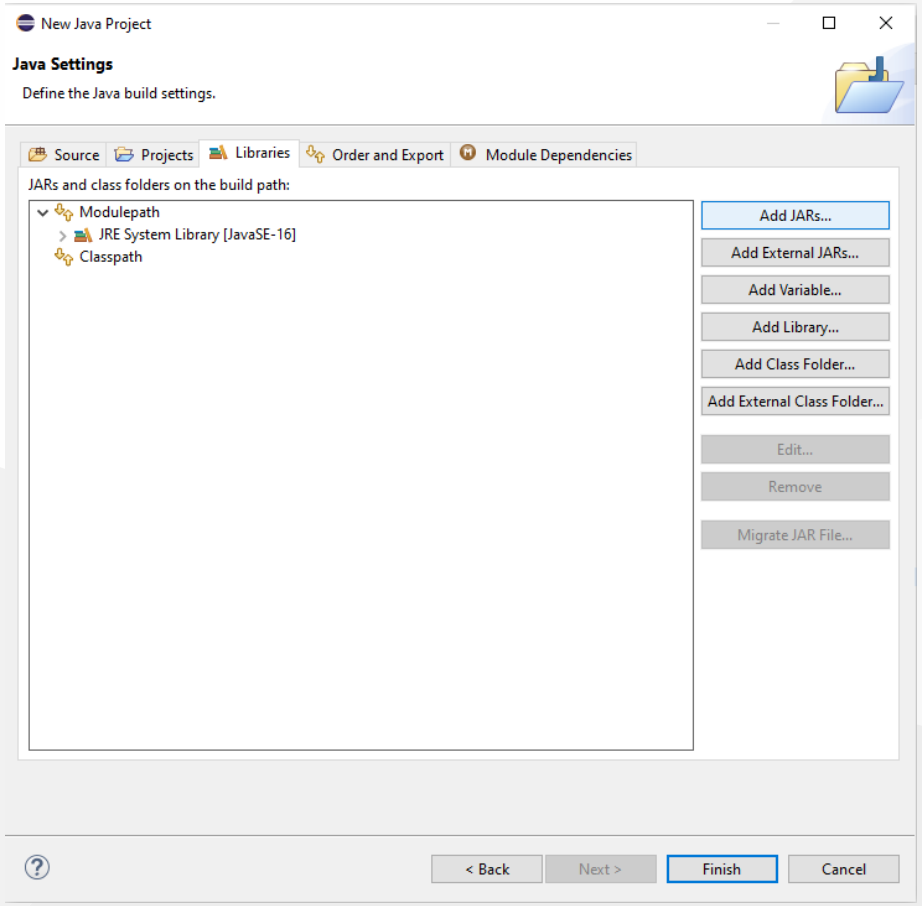


Shared Library Development - (Eclipse Java Jar Library)-56



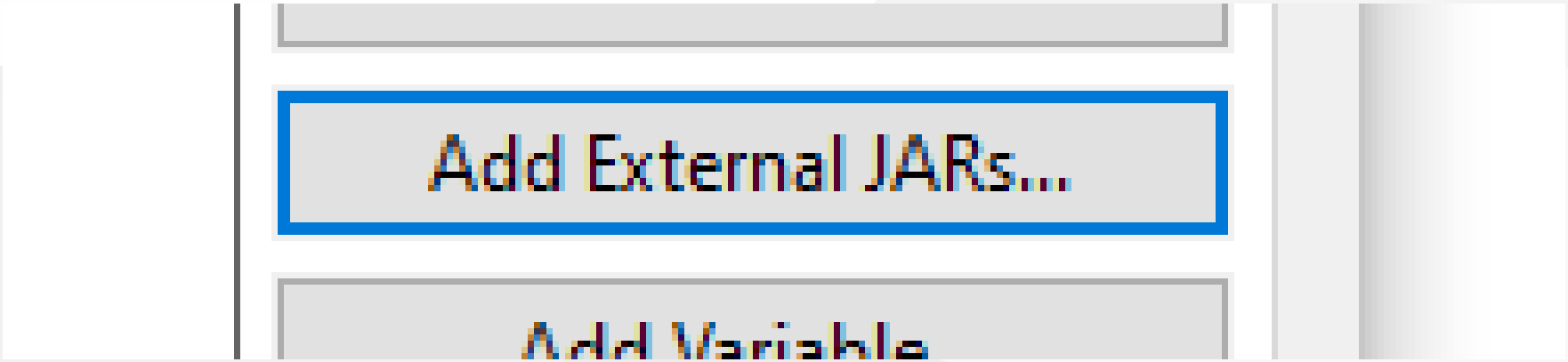
Shared Library Development - (Eclipse Java Jar Library)-57

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



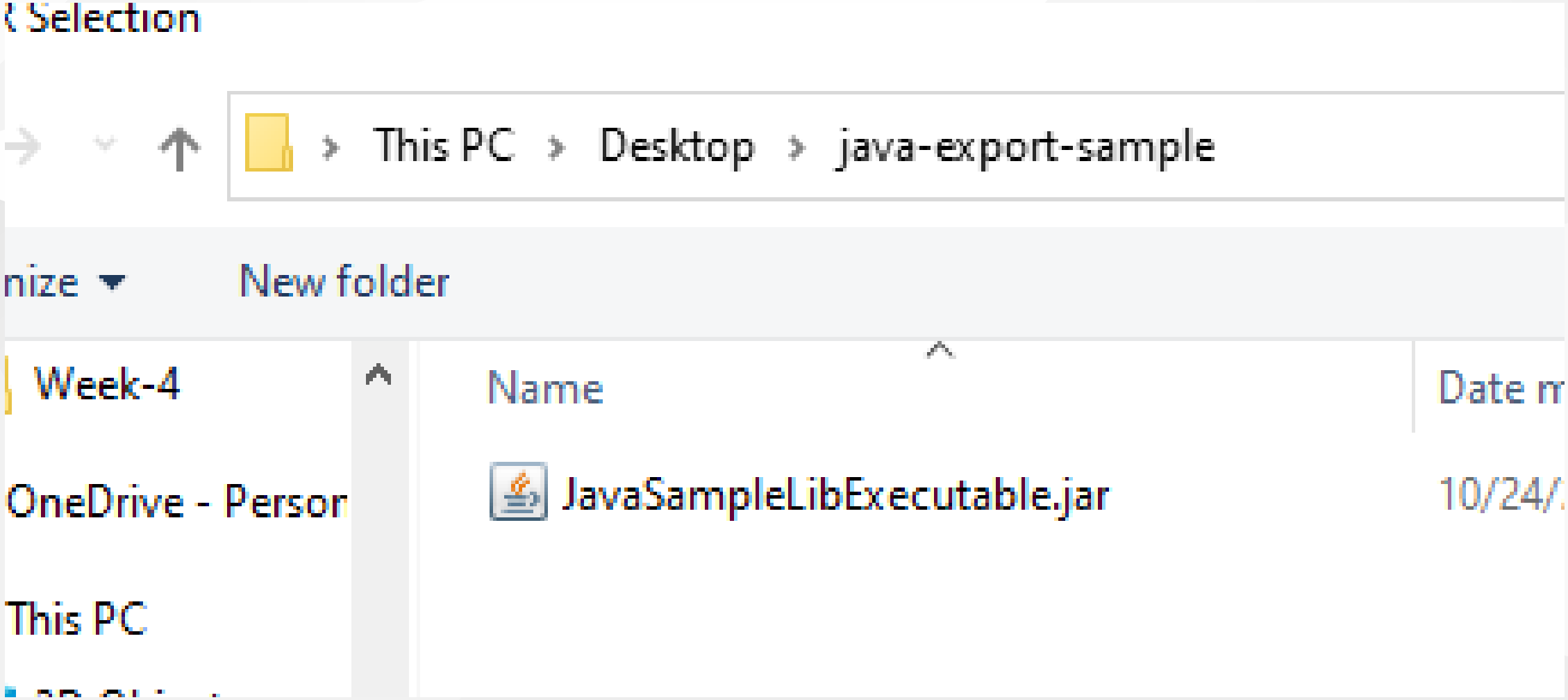
Shared Library Development - (Eclipse Java Jar Library)-58

- Select Add External JARs...



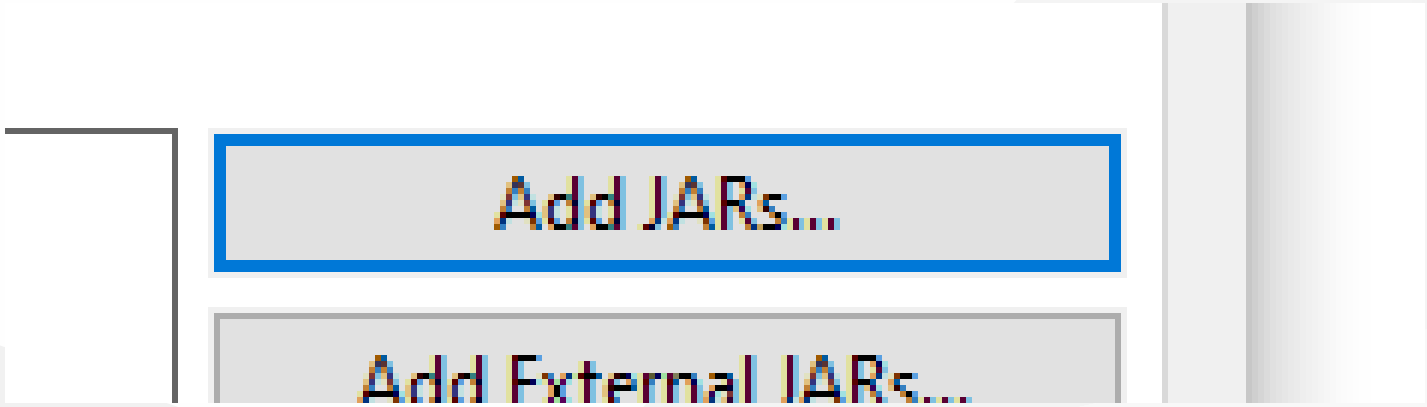
Shared Library Development - (Eclipse Java Jar Library)-59

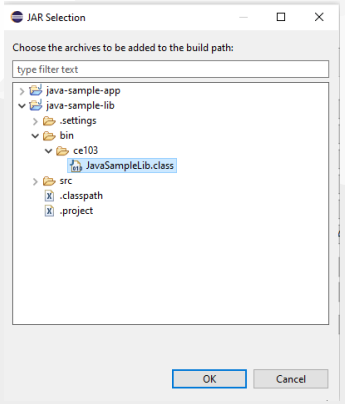
- Open Exported jar folder and select



Shared Library Development - (Eclipse Java Jar Library)-60

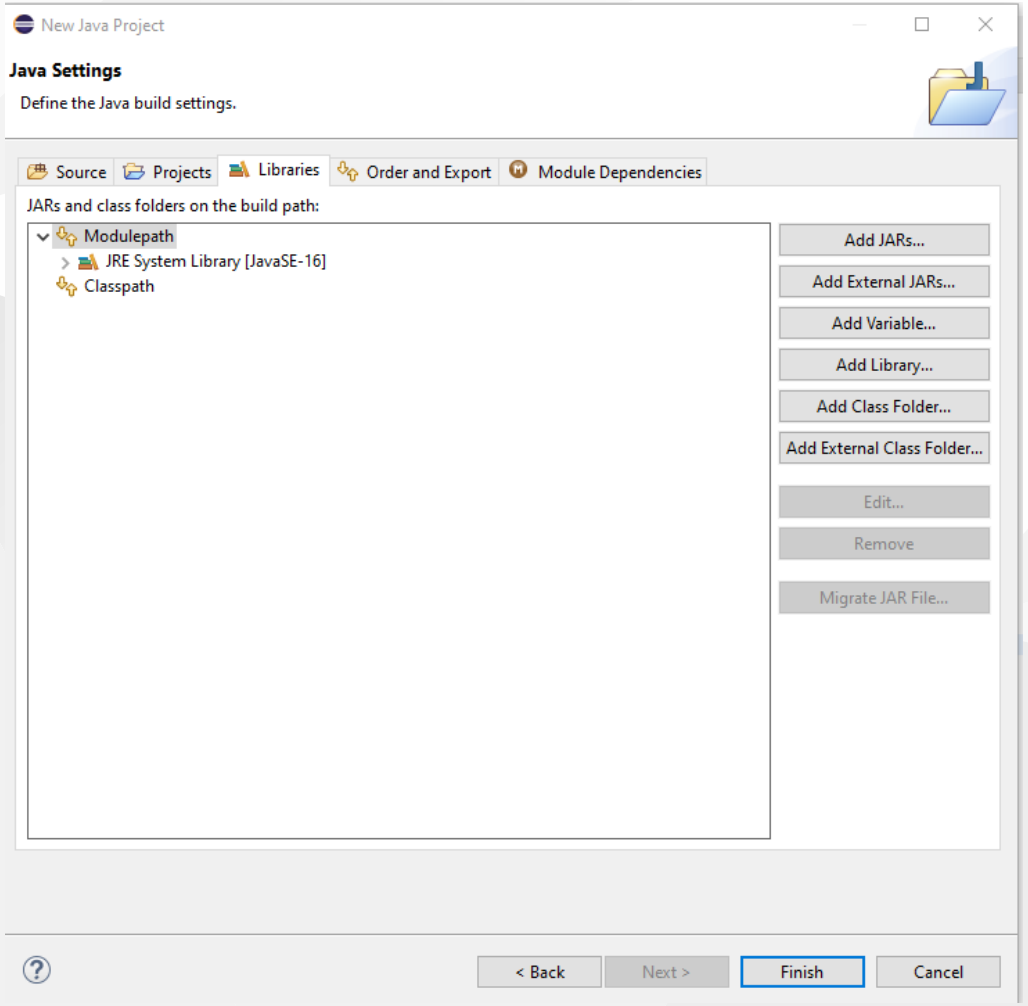
- Or we can select by Add jar from current workspace





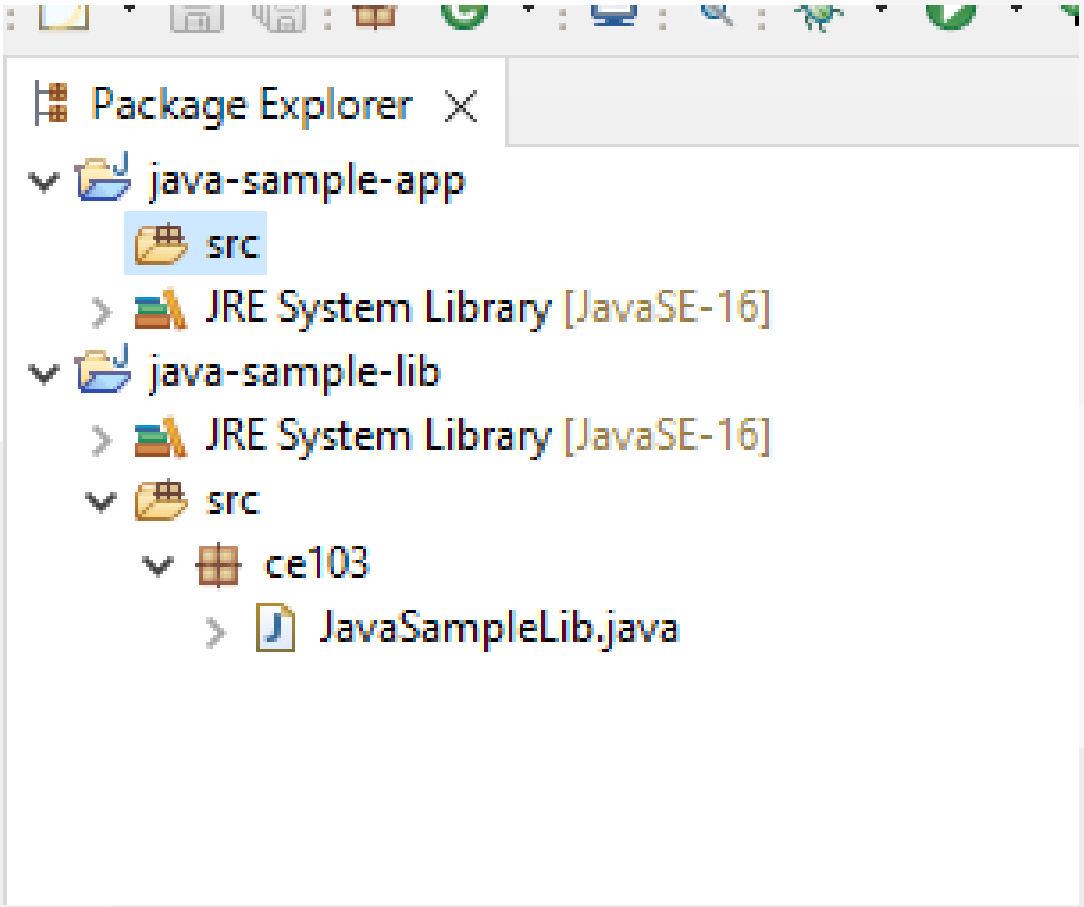
Shared Library Development - (Eclipse Java Jar Library)-61

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



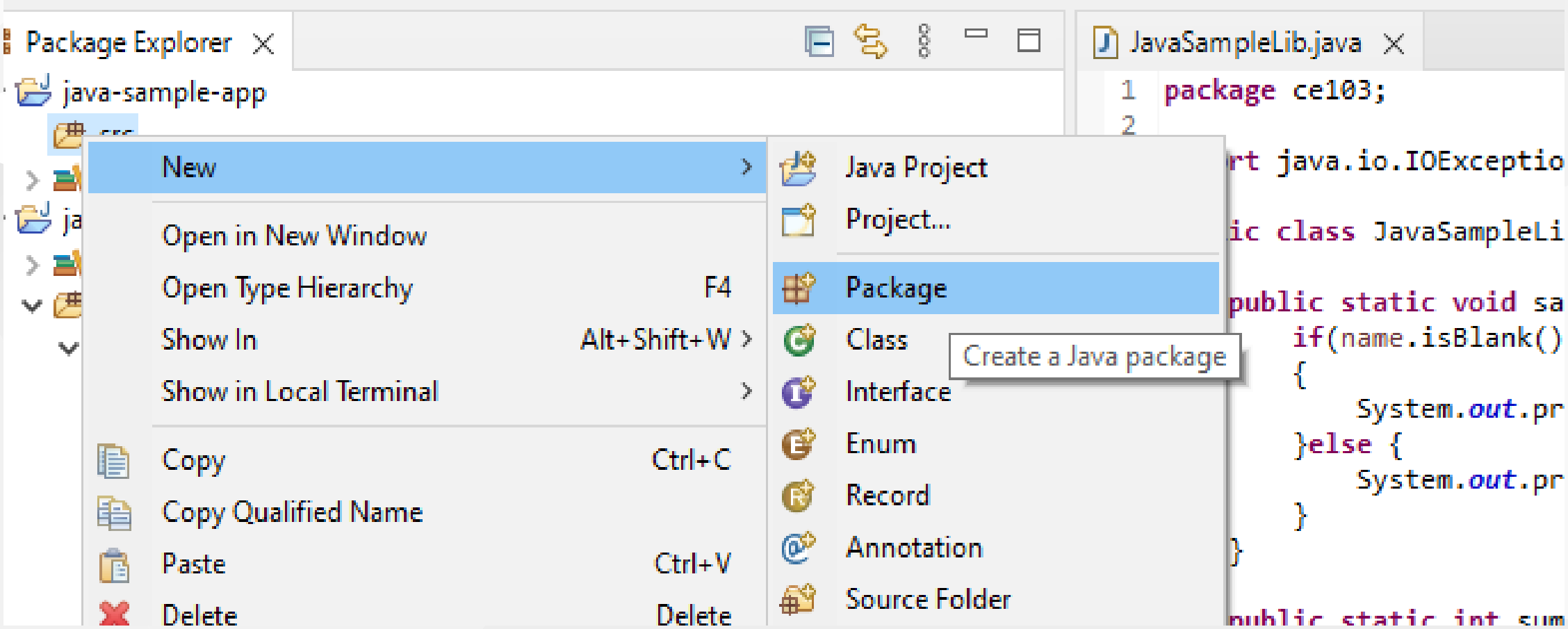
Shared Library Development - (Eclipse Java Jar Library)-62

- we will have the following project

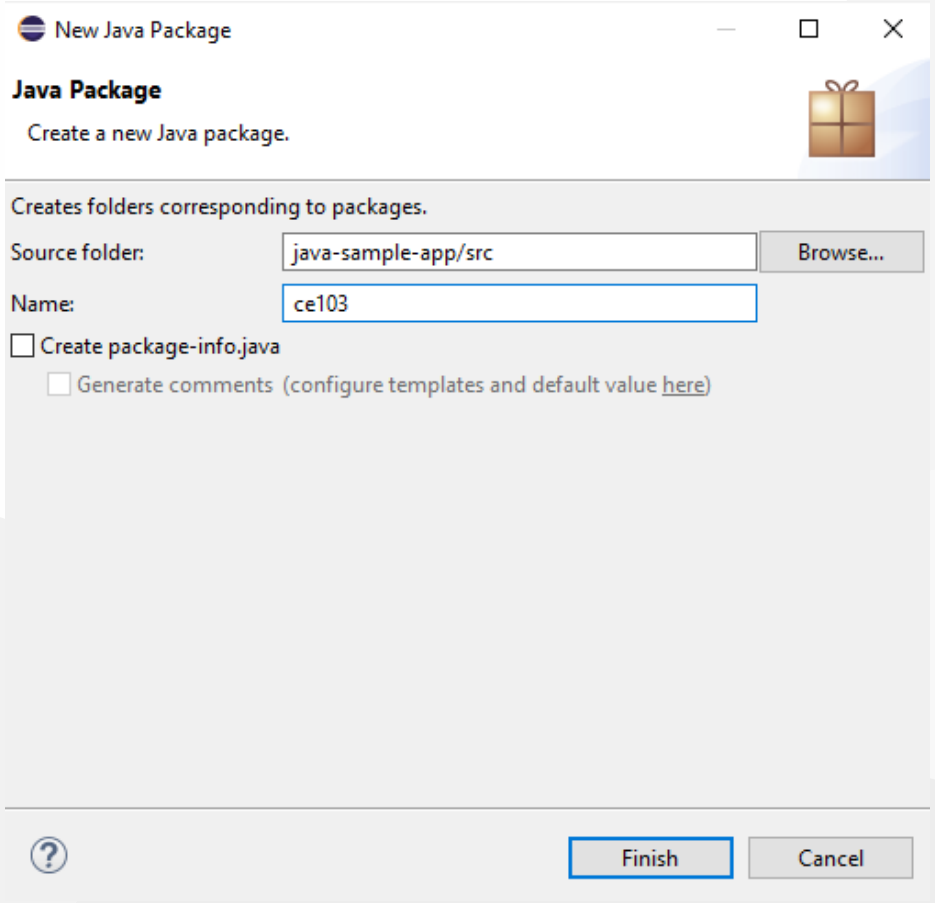


Shared Library Development - (Eclipse Java Jar Library)-63

- lets create a package

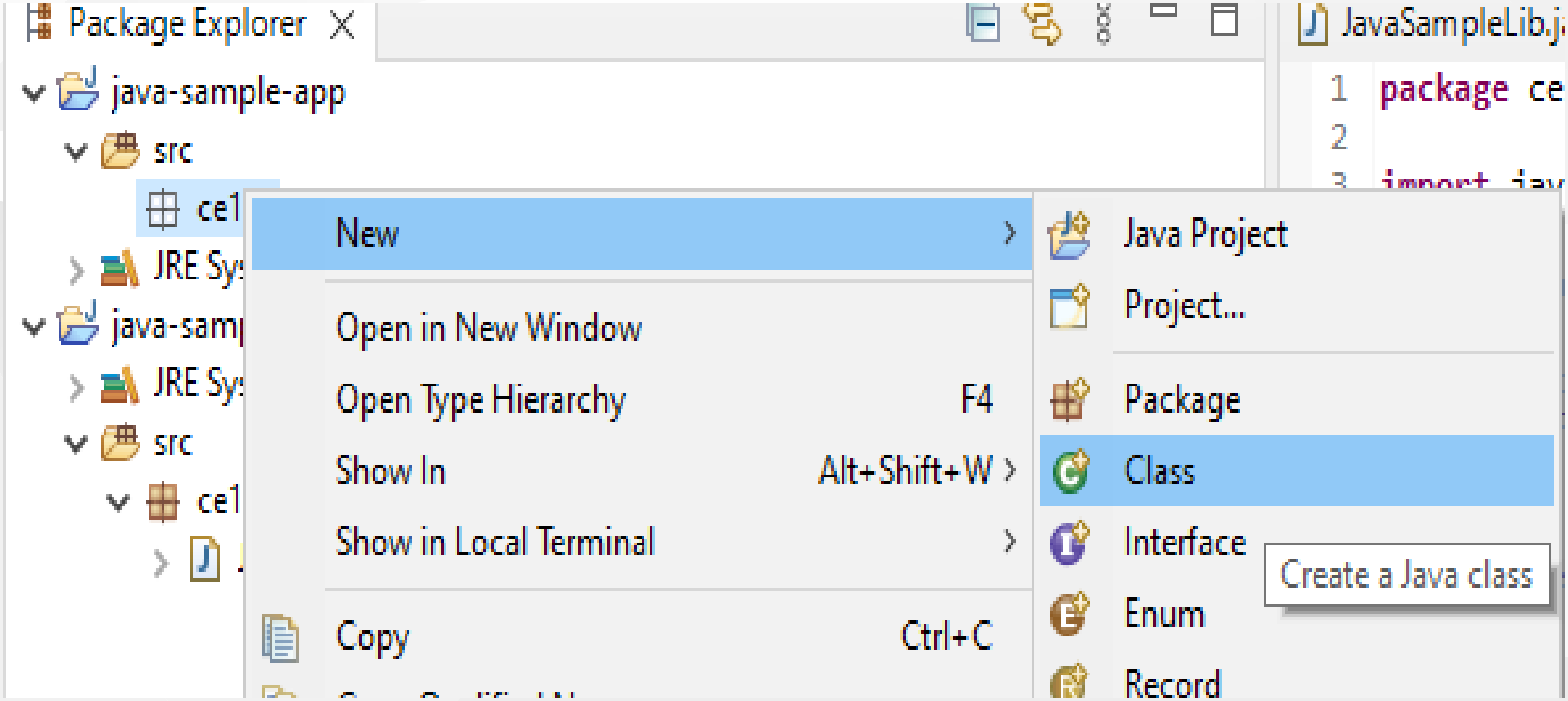


Shared Library Development - (Eclipse Java Jar Library)-64



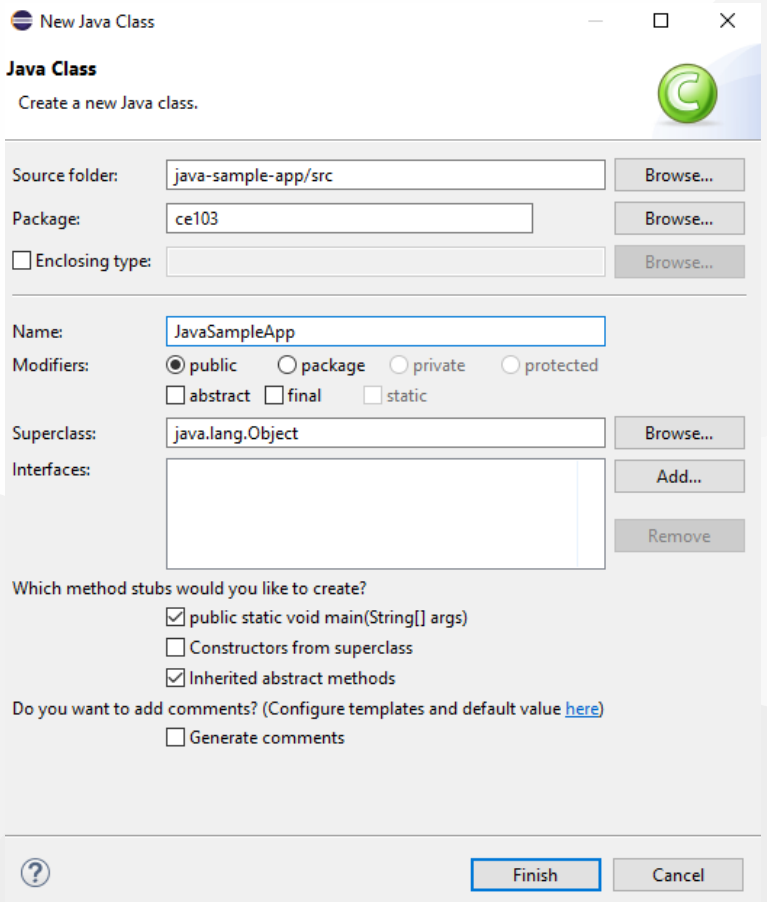
Shared Library Development - (Eclipse Java Jar Library)-65

- and lets create a main class for our application

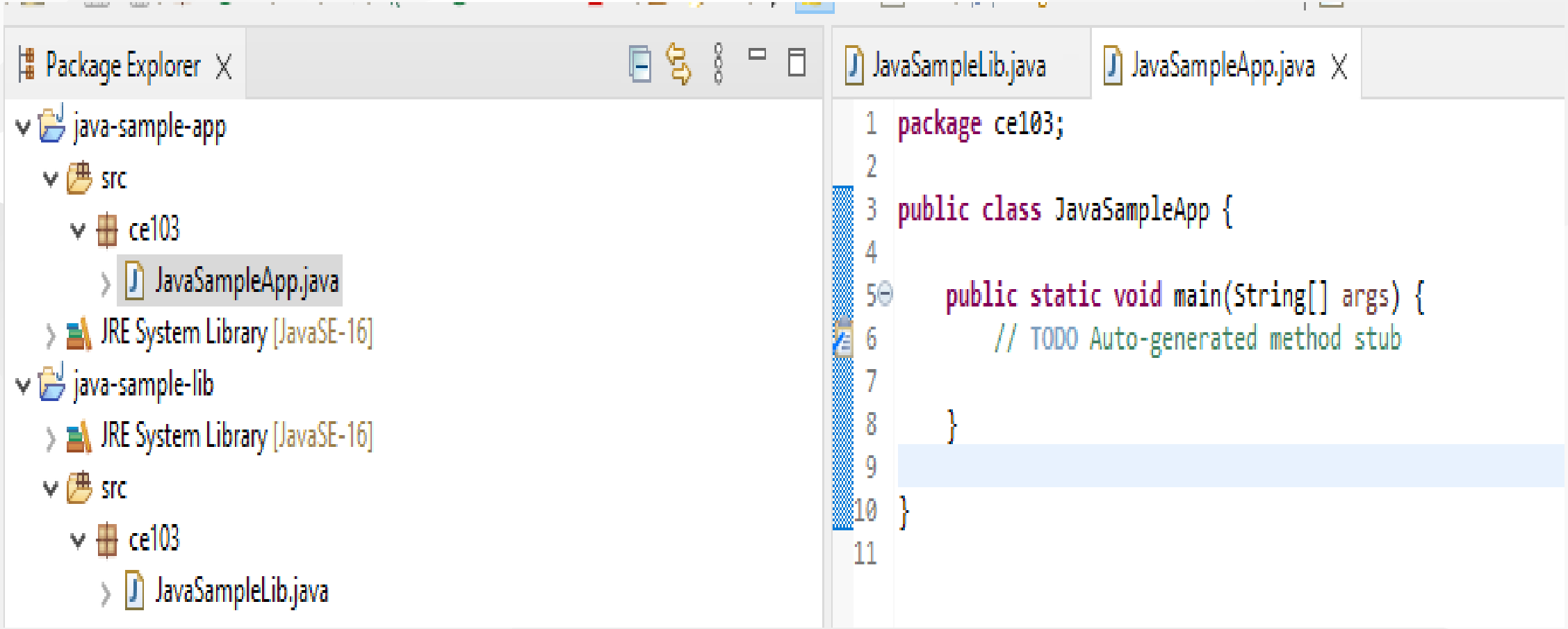


Shared Library Development - (Eclipse Java Jar Library)-66

- check create main function

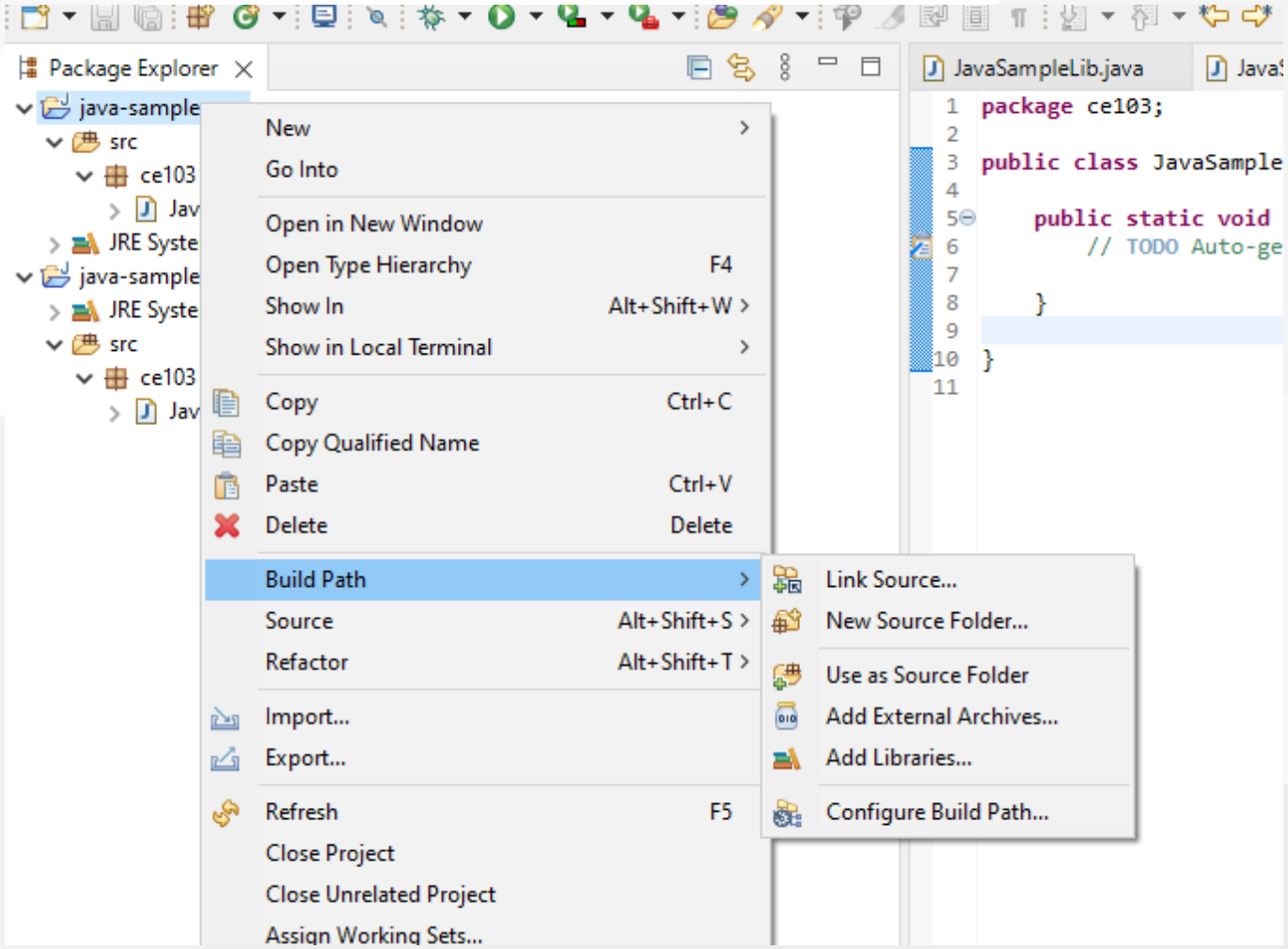


Shared Library Development - (Eclipse Java Jar Library)-67



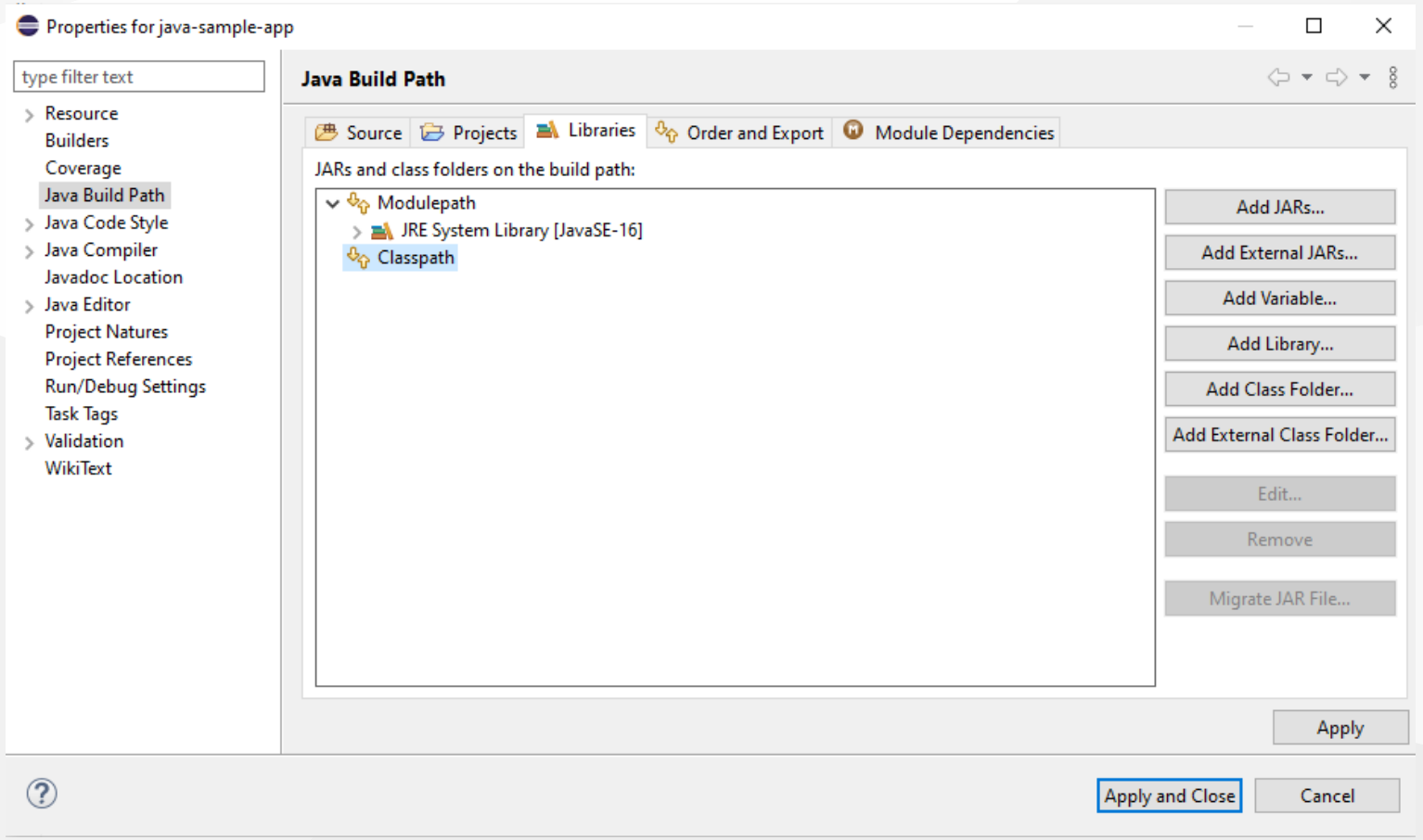
Shared Library Development - (Eclipse Java Jar Library)-68

- right click to project and add reference



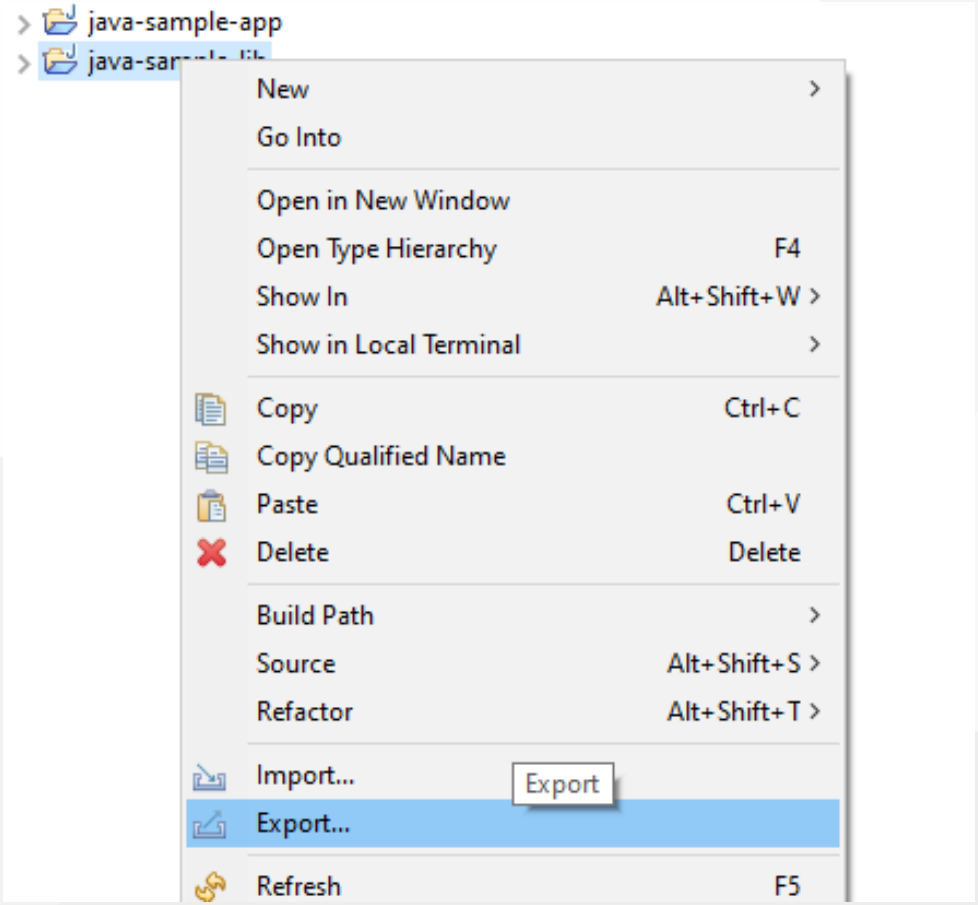
Shared Library Development - (Eclipse Java Jar Library)-69

- you can enter same configurations from project properties



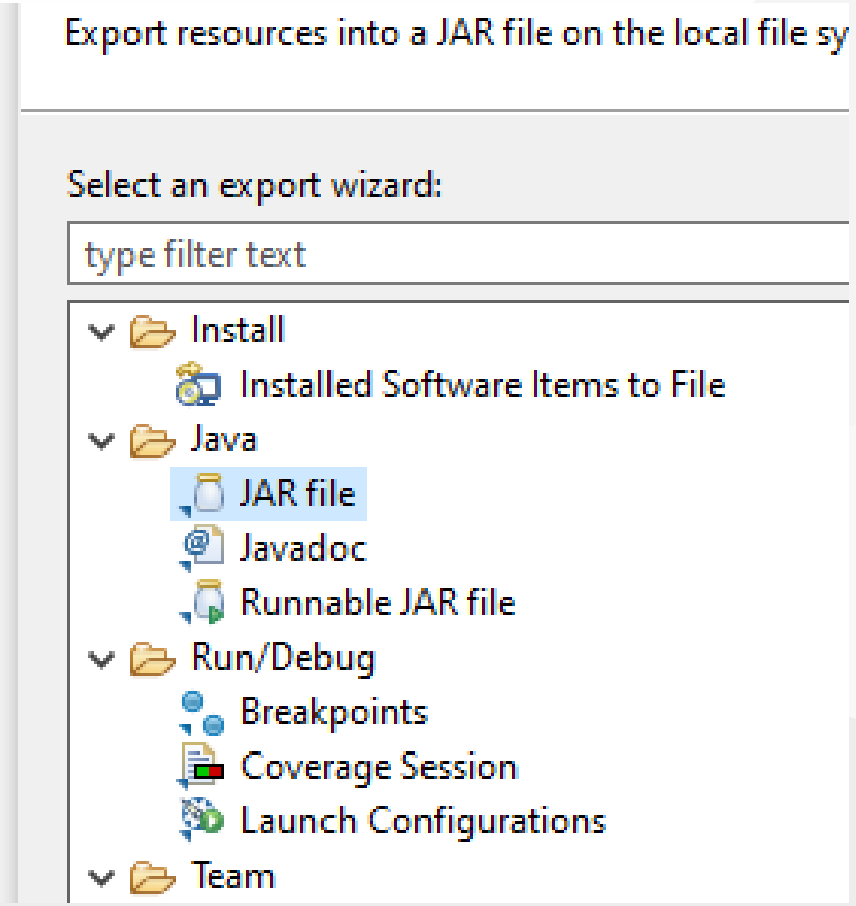
Shared Library Development - (Eclipse Java Jar Library)-70

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



Shared Library Development - (Eclipse Java Jar Library)-71

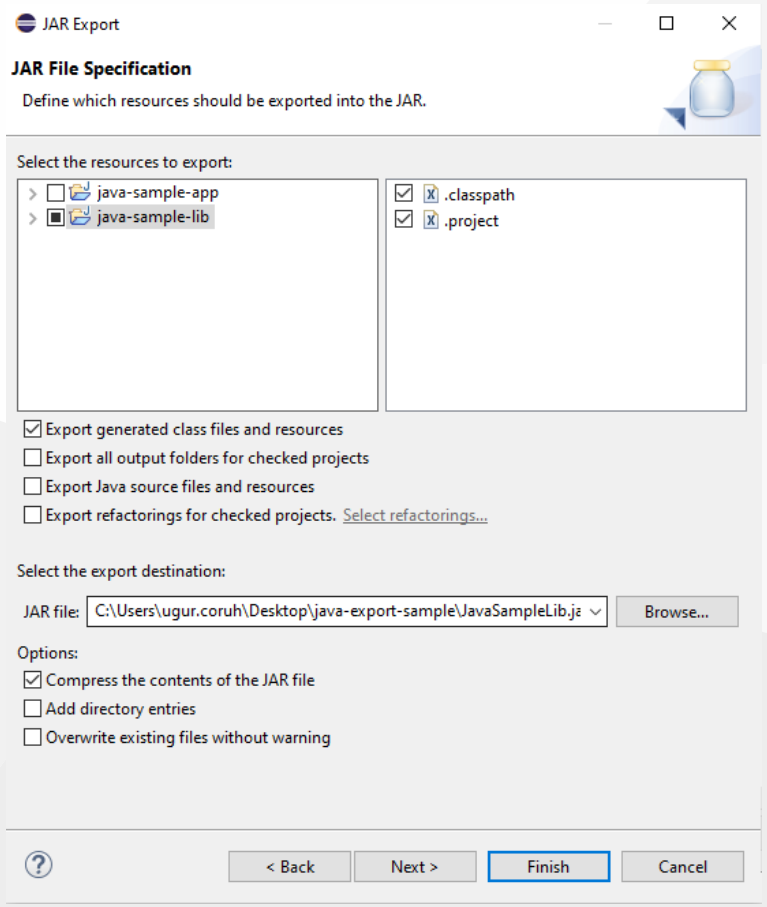
Select JAR file



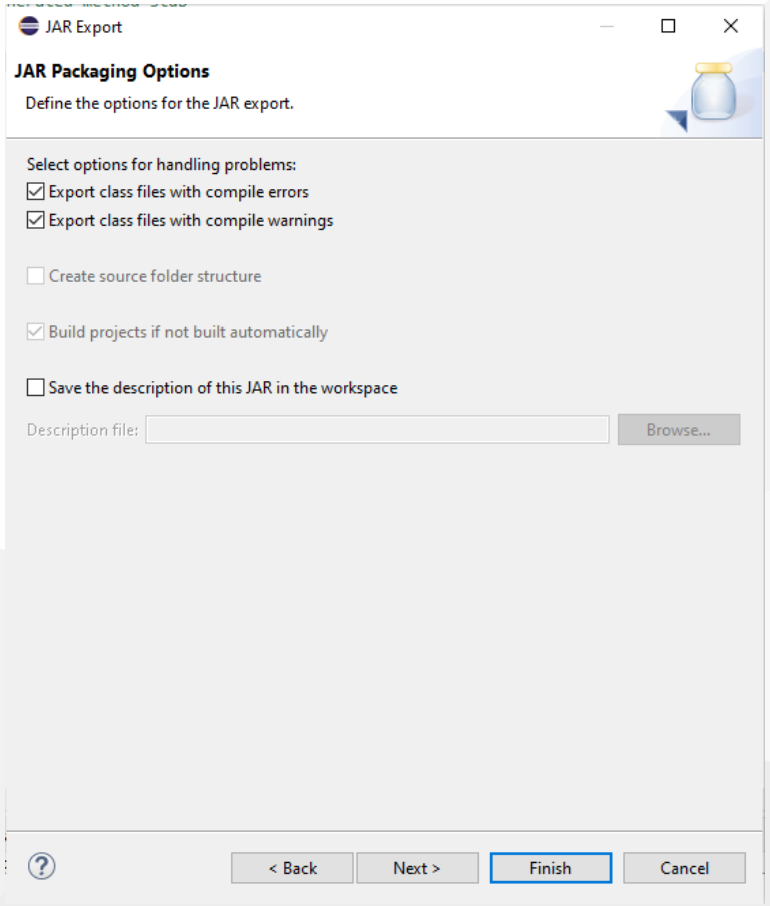
Shared Library Development - (Eclipse Java Jar Library)-72

we configured output as

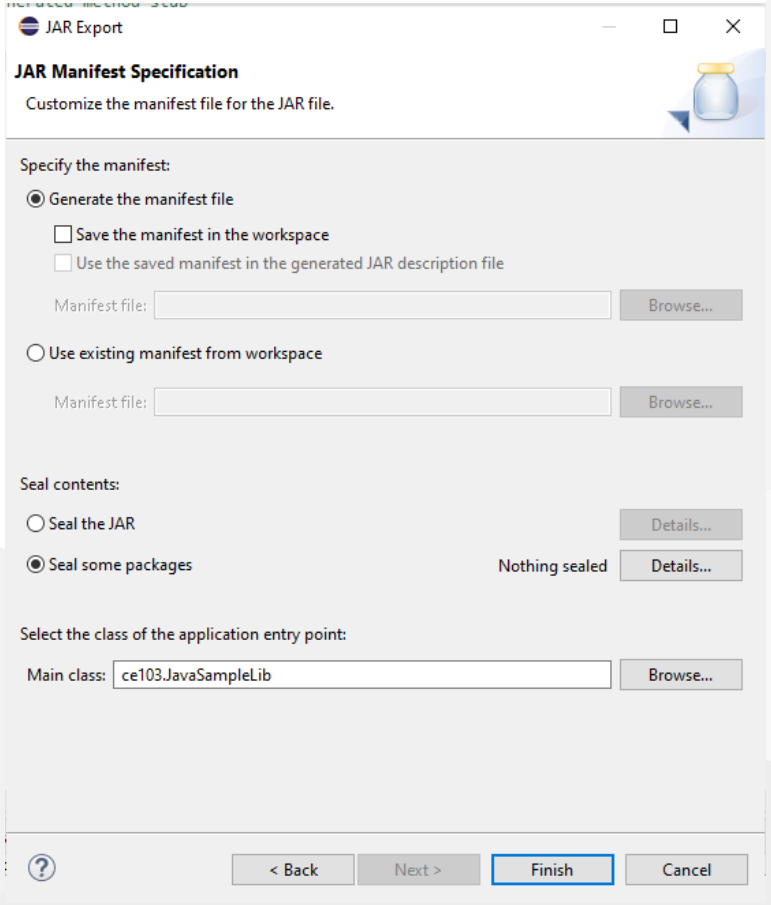
```
C:\Users\ugur.coruh\Desktop\java-export-sample\JavaSampleLib.jar
```



Shared Library Development - (Eclipse Java Jar Library)-73

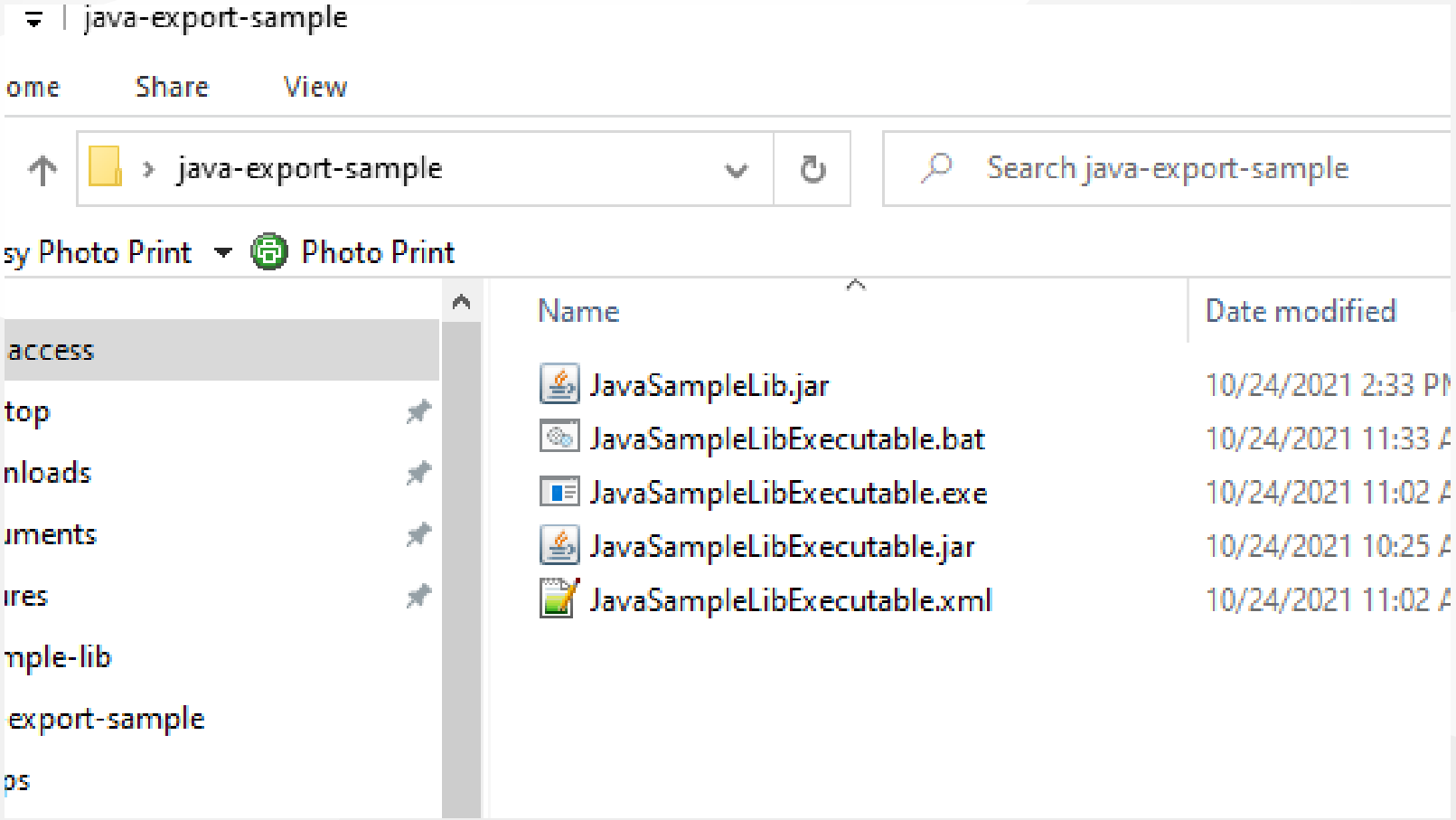


Shared Library Development - (Eclipse Java Jar Library)-74



Shared Library Development - (Eclipse Java Jar Library)-75

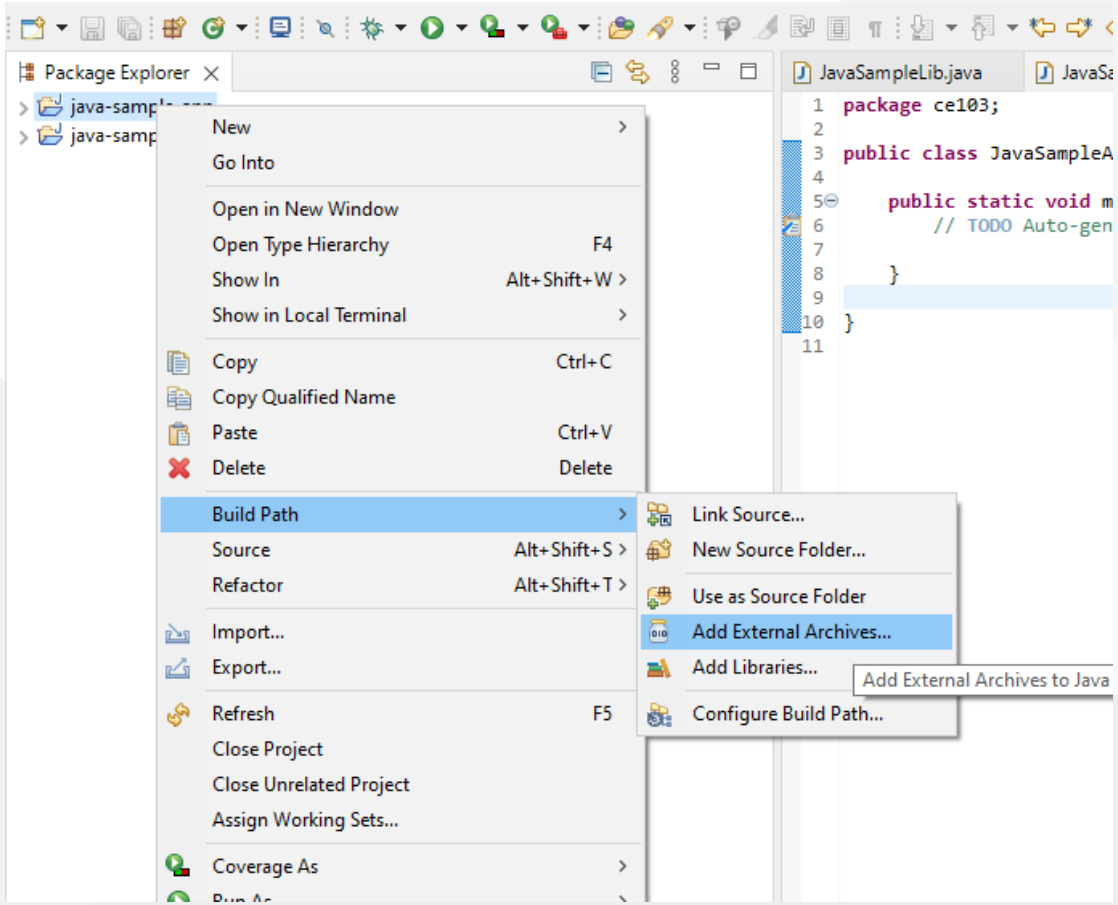
In the same export folder now we have JavaSampleLib.jar



Shared Library Development - (Eclipse Java Jar Library)-76

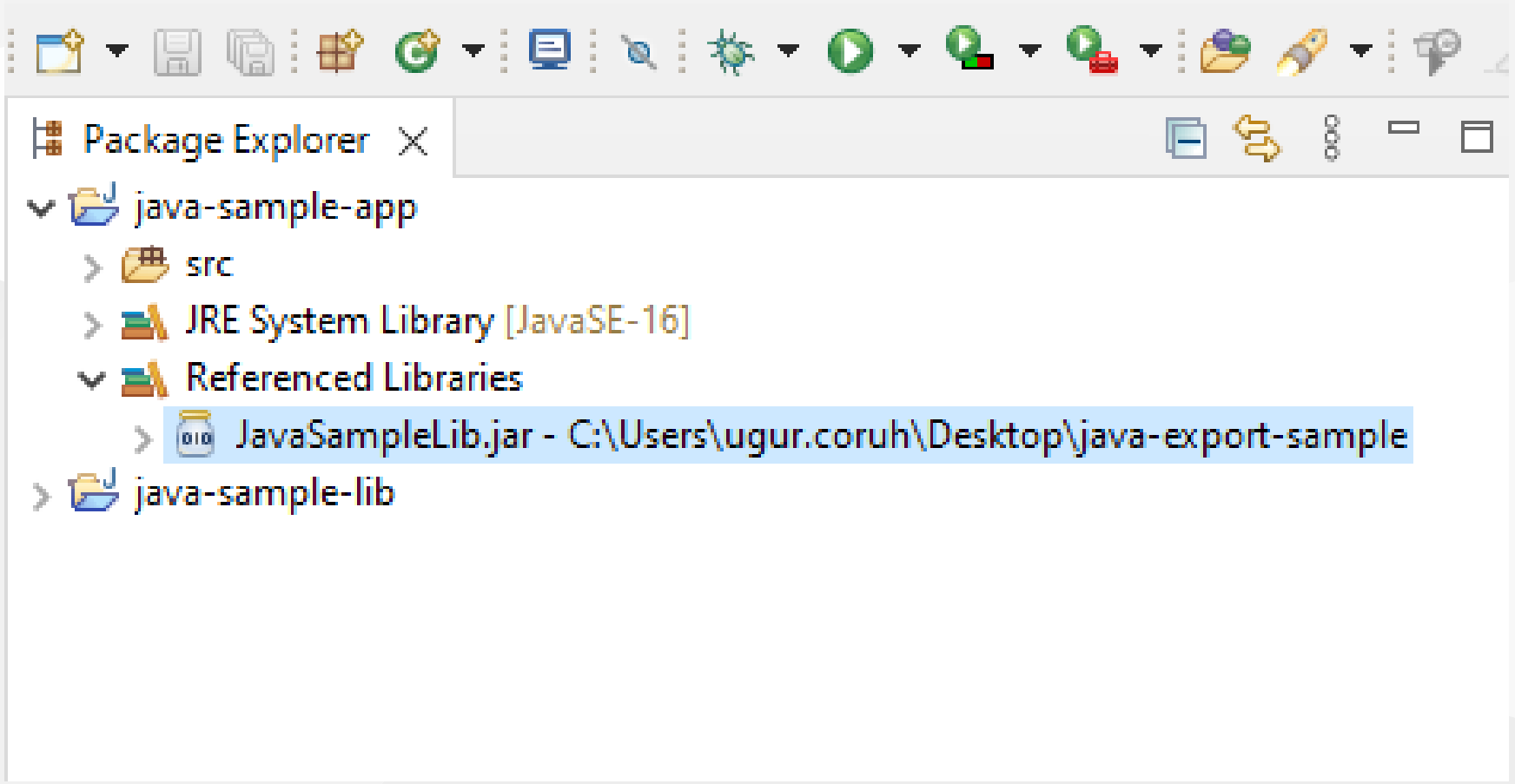
return back to java-sample-app and then add this jar file to our project

Build Path->Add External Archives



Shared Library Development - (Eclipse Java Jar Library)-77

you will see its added to reference libraries



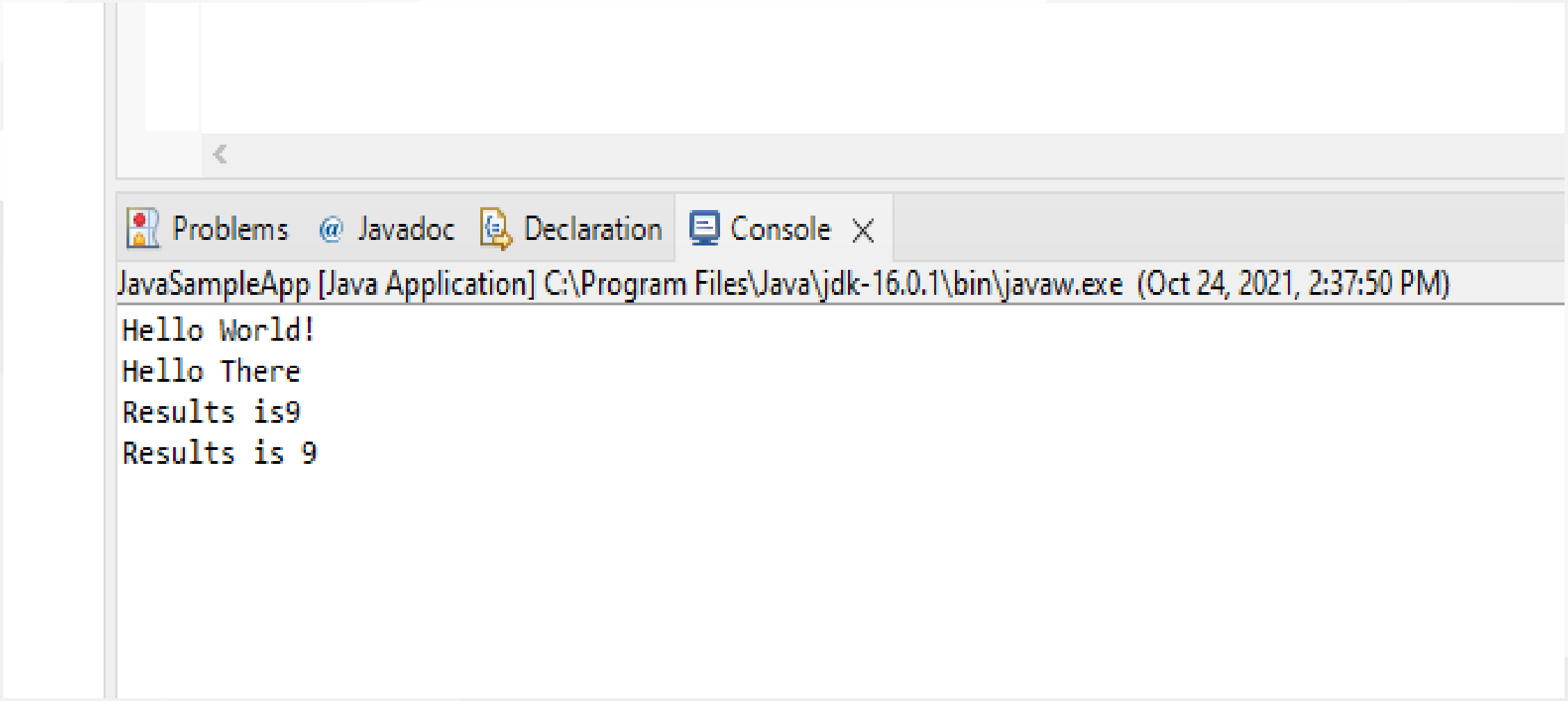
Shared Library Development - (Eclipse Java Jar Library)-78

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();
        }

    }

}
```

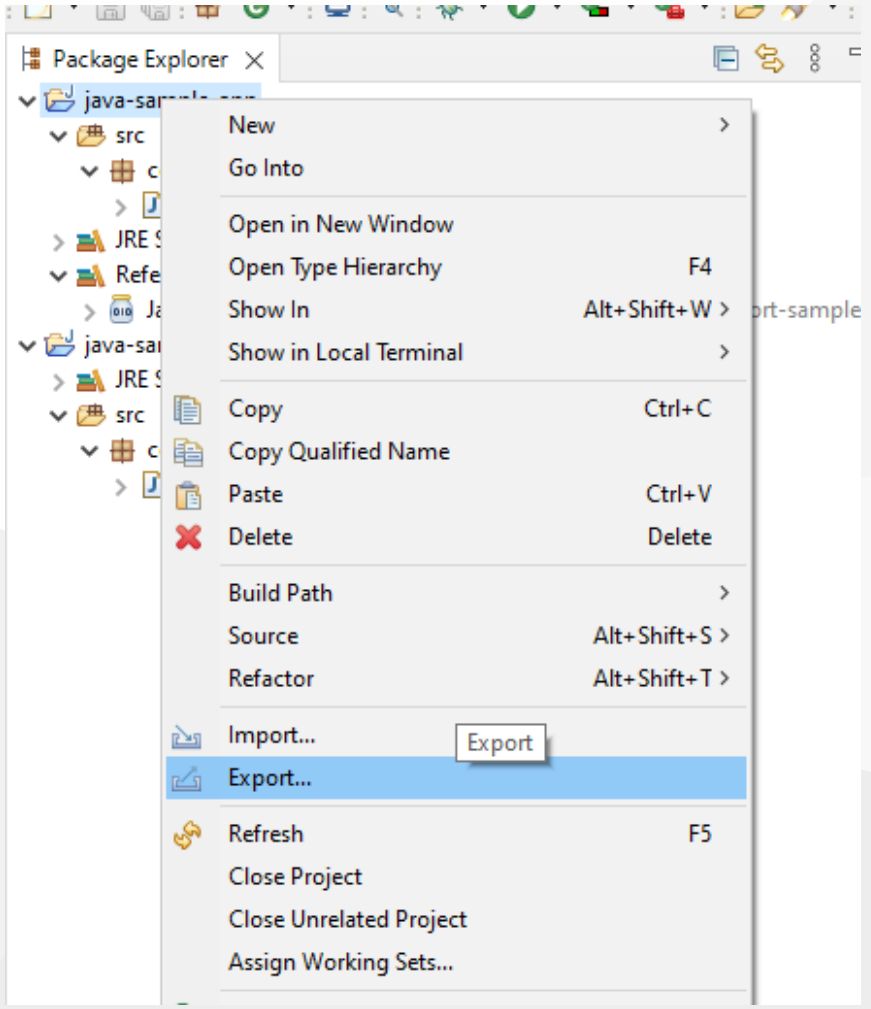
Shared Library Development - (Eclipse Java Jar Library)-79

When we run application we will see similar output



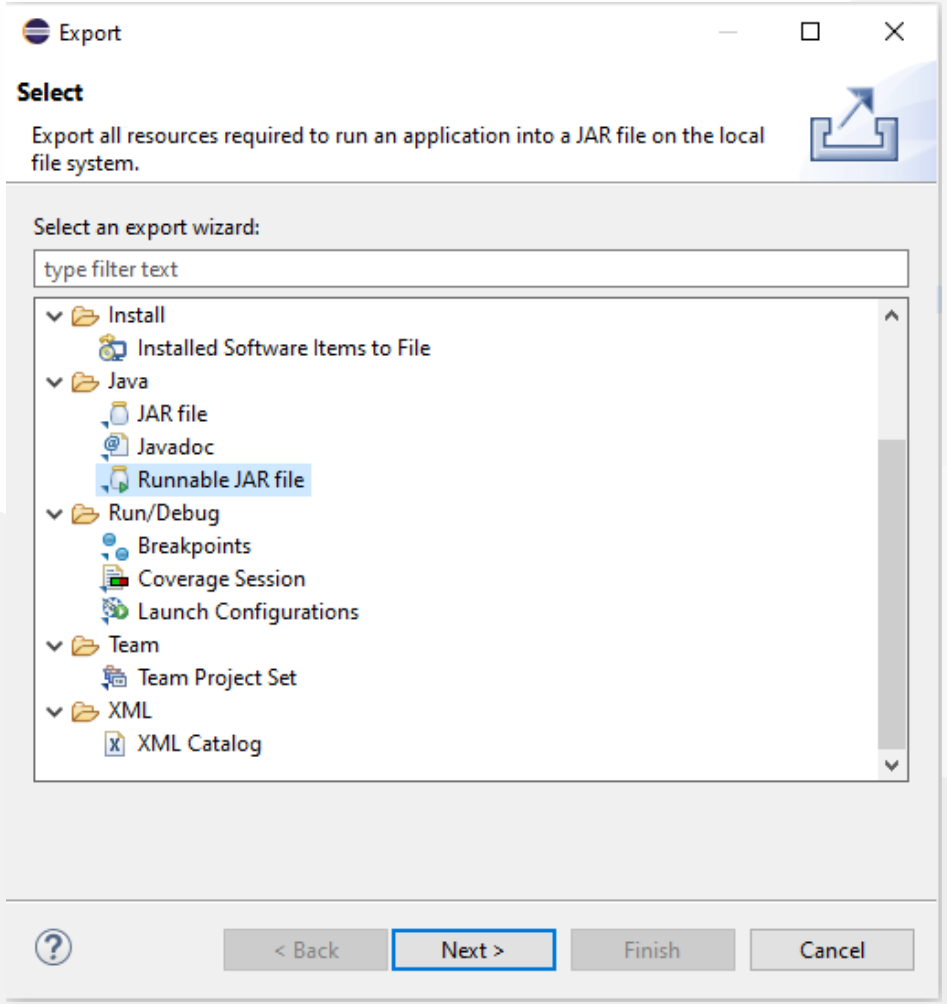
Shared Library Development - (Eclipse Java Jar Library)-80

Lets export this application with its dependent library



Shared Library Development - (Eclipse Java Jar Library)-81

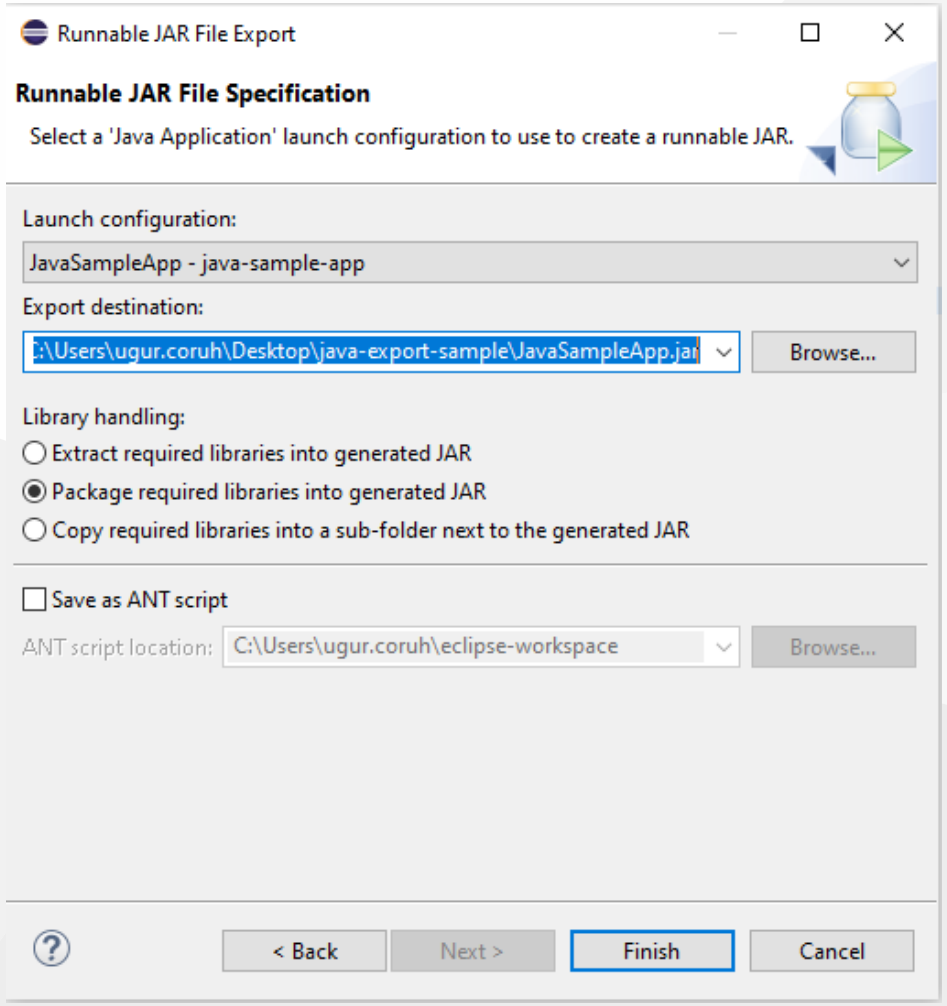
Select runnable jar



Shared Library Development - (Eclipse Java Jar Library)-82

Set Launch configuration and Export destination

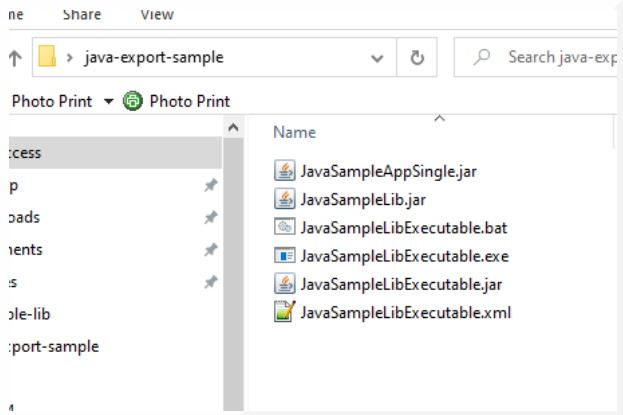
```
C:\Users\ugur.coruh\Desktop\java-export-sample\JavaSampleAppSingle.jar
```



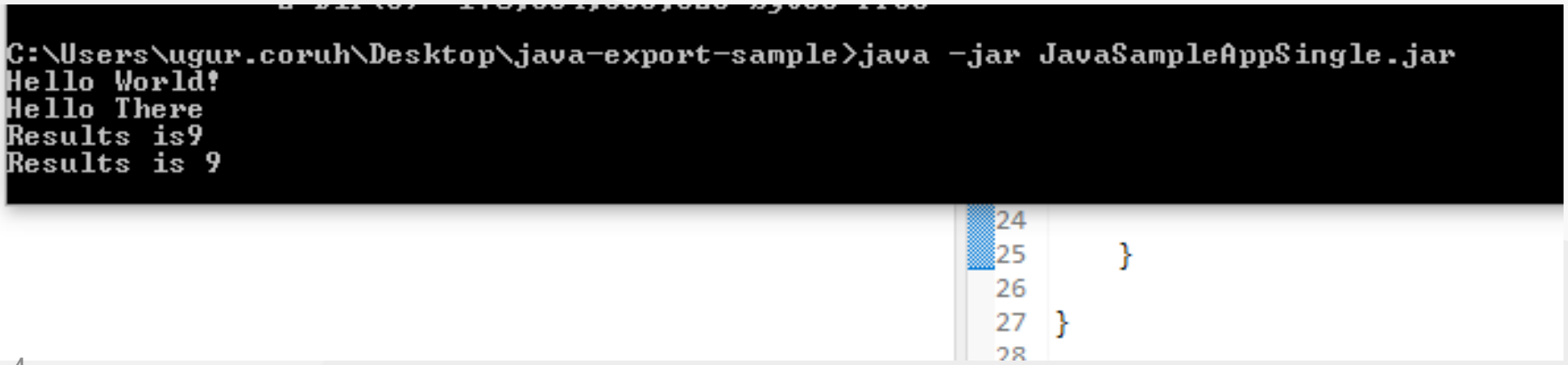
Shared Library Development - (Eclipse Java Jar Library)-83

In this option we will have single jar file

In the export folder we do not see reference libraries



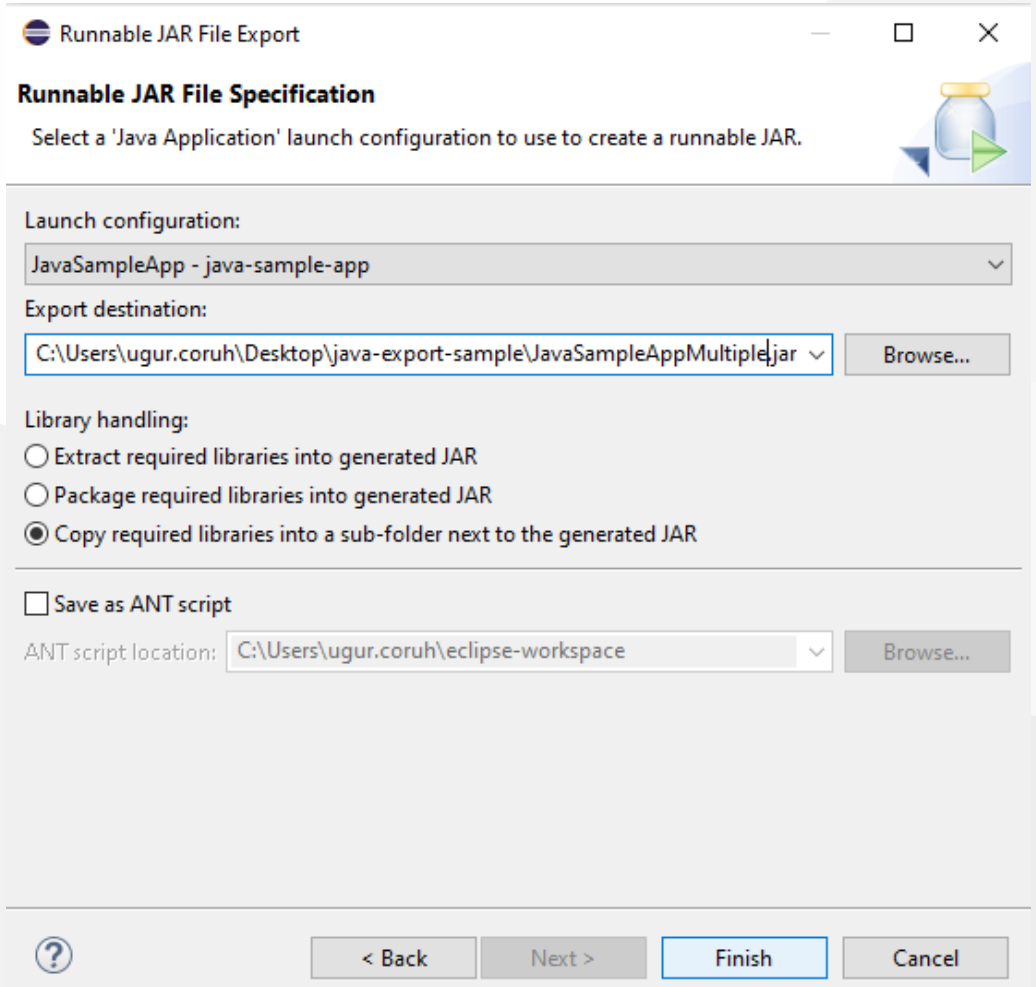
and we can run with command line



Shared Library Development - (Eclipse Java Jar Library)-84

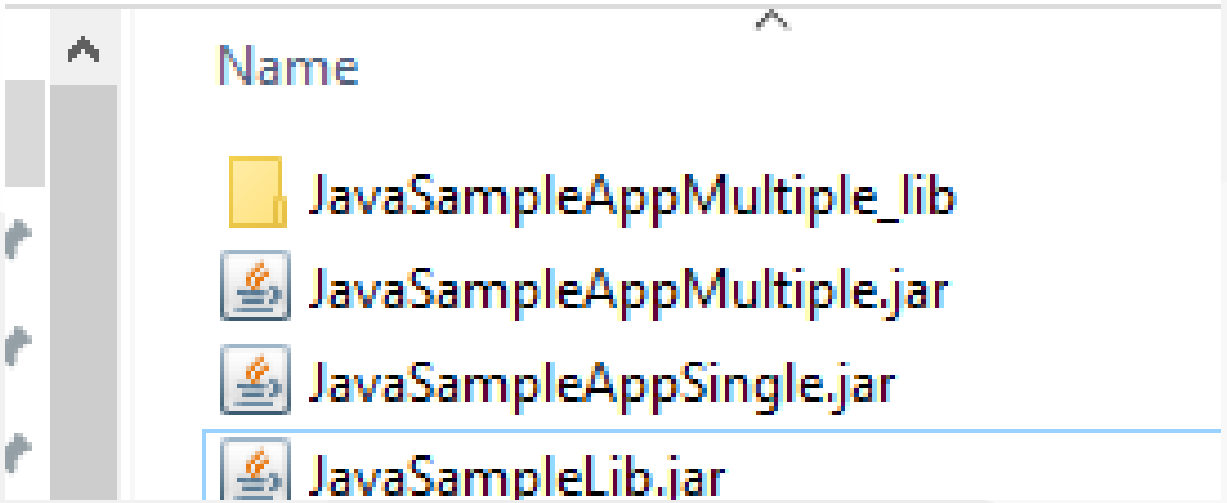
only change copy required libraries setting and then give a new name for new jar file and export

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



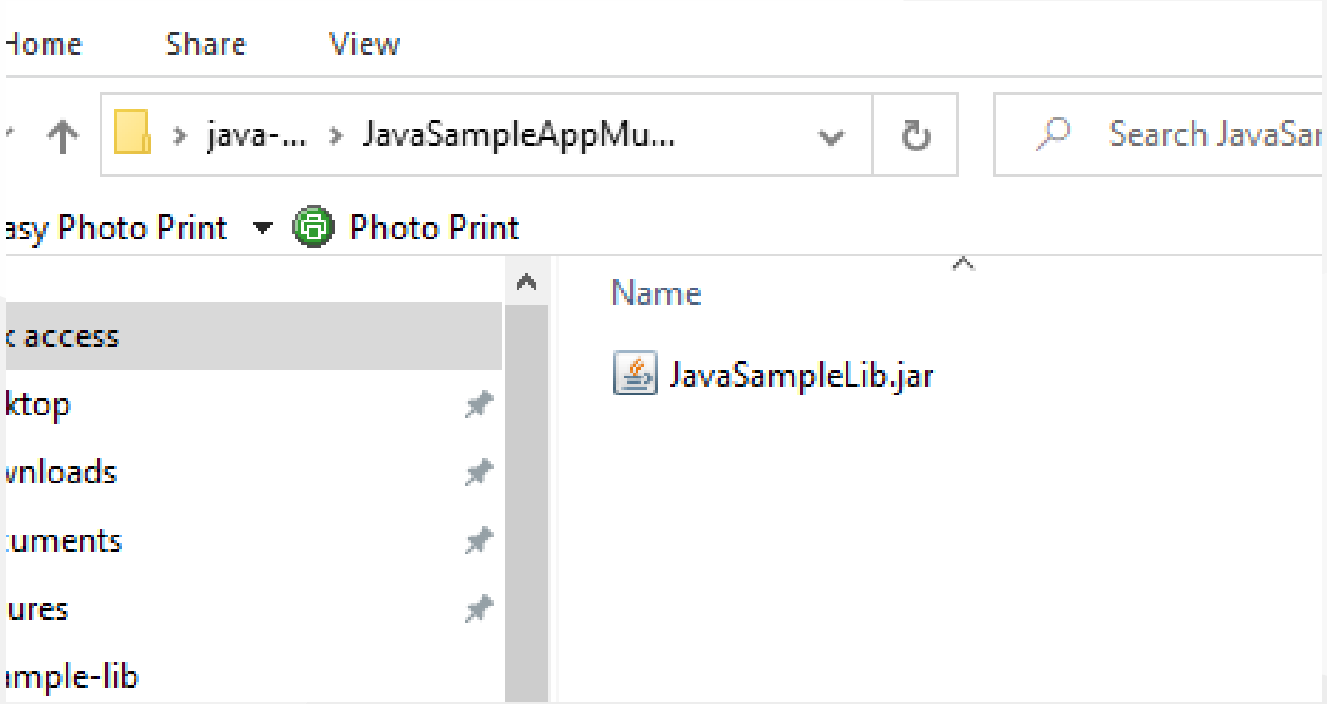
Shared Library Development - (Eclipse Java Jar Library)-85

now we have a folder that contains our libraries referenced



Shared Library Development - (Eclipse Java Jar Library)-86

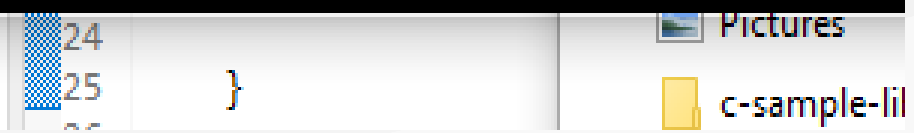
in this file we can find our library



Shared Library Development - (Eclipse Java Jar Library)-87

if we test our application we will see it will work

```
C:\Users\ugur.coruh\Desktop\java-export-sample>java -jar JavaSampleAppMultiple.jar
Hello World!
Hello There
Results is9
Results is 9
```



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

```
C:\Users\ugur.coruh\Desktop\java-export-sample>java -jar JavaSampleAppMultiple.jar
Hello World!
Exception in thread "main" java.lang.NoClassDefFoundError: ce103/JavaSampleLib
        at ce103.JavaSampleApp.main(JavaSampleApp.java:12)
Caused by: java.lang.ClassNotFoundException: ce103.JavaSampleLib
        at java.base/jdk.internal.loader.BuiltinClassLoader.loadClass(BuiltinClassLoader.java:636)
        at java.base/jdk.internal.loader.ClassLoaders$AppClassLoader.loadClass(ClassLoaders.java:182)
        at java.base/java.lang.ClassLoader.loadClass(ClassLoader.java:519)
        ... 1 more

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

Application Testing

- C
- C++
- C#
- Java

Unit Test Development

Wikipedia Unit Test Library List for Each Language

https://en.wikipedia.org/wiki/List_of_unit_testing_frameworks

Visual Studio Community Edition

C Unit Tests

Visual Studio Community Edition - C Unit Tests

- There is no direct C source testing but with additional frameworks. Visual Studio can test C sources.
- You can check the following entry
  - https://stackoverflow.com/questions/65820/unit-testing-c-code
- Recommended framework is Check
  - https://libcheck.github.io/check/web/install.html
  - https://github.com/libcheck/check/releases

Visual Studio Community Edition

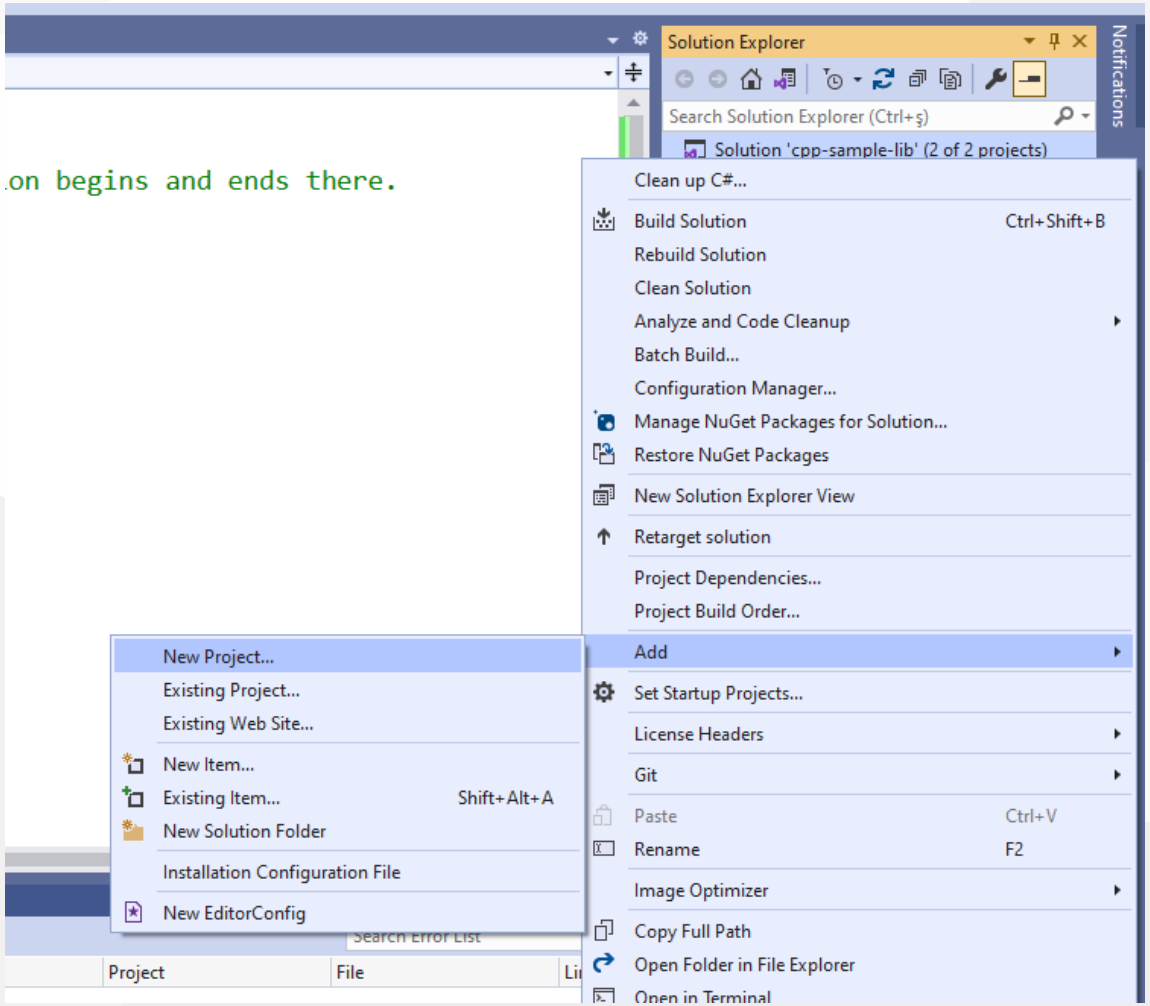
C++ Unit Tests

Visual Studio Community Edition - C++ Unit Tests-1

- C/C++ için birim testleri yazma - Visual Studio (Windows) | Microsoft Docs

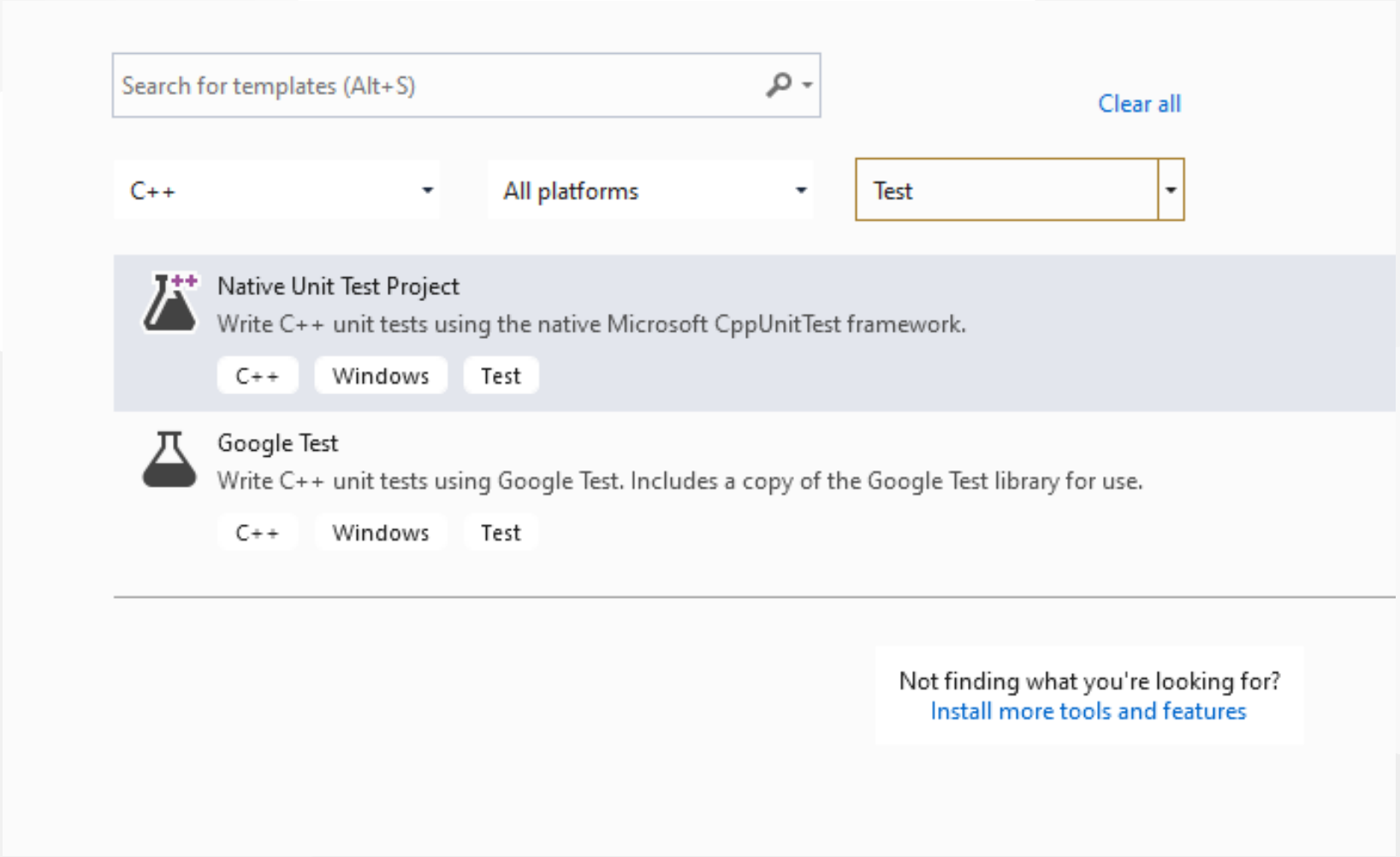
Visual Studio Community Edition - C++ Unit Tests-2

- Use cpp-sample-lib project and add



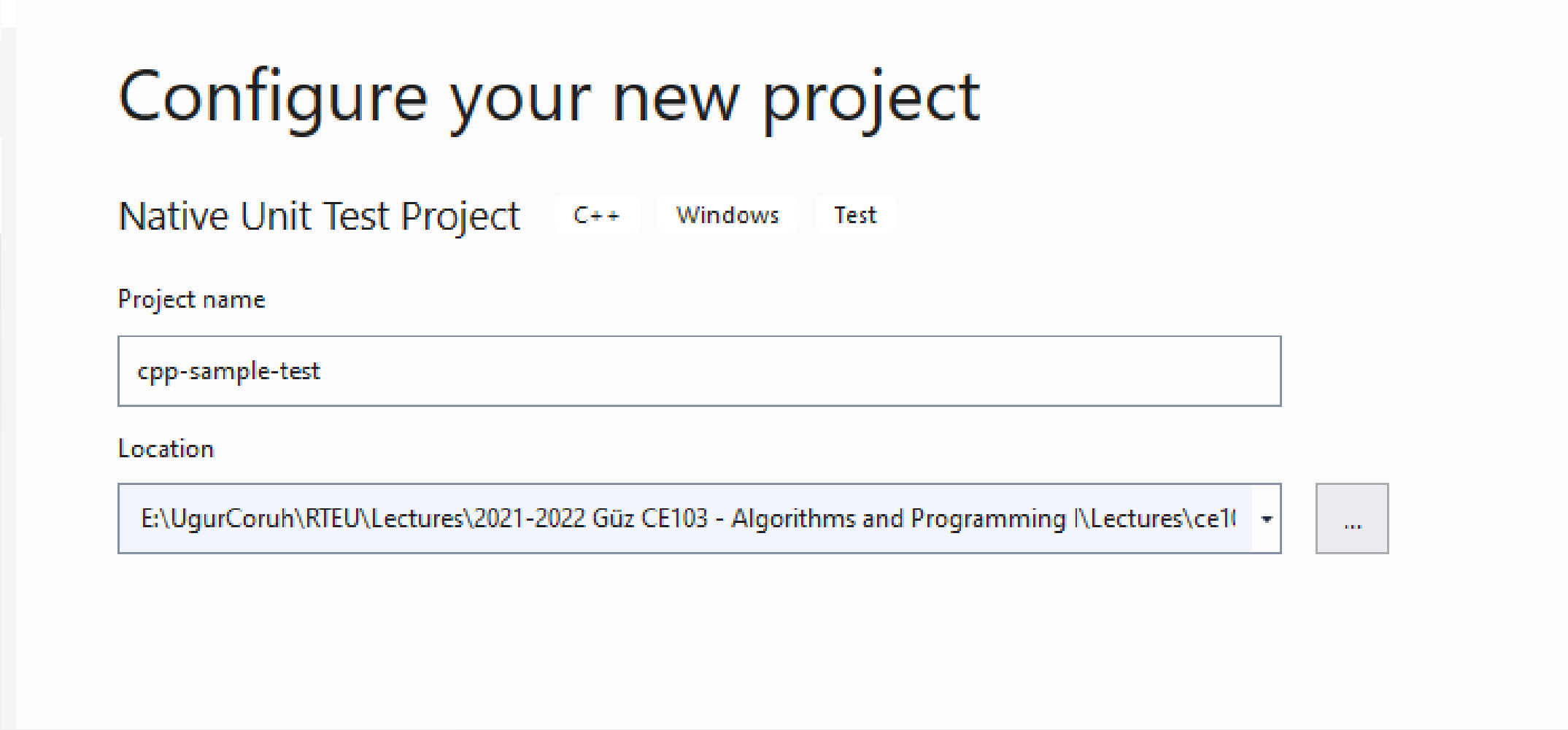
Visual Studio Community Edition - C++ Unit Tests-3

- Select Native Unit Test



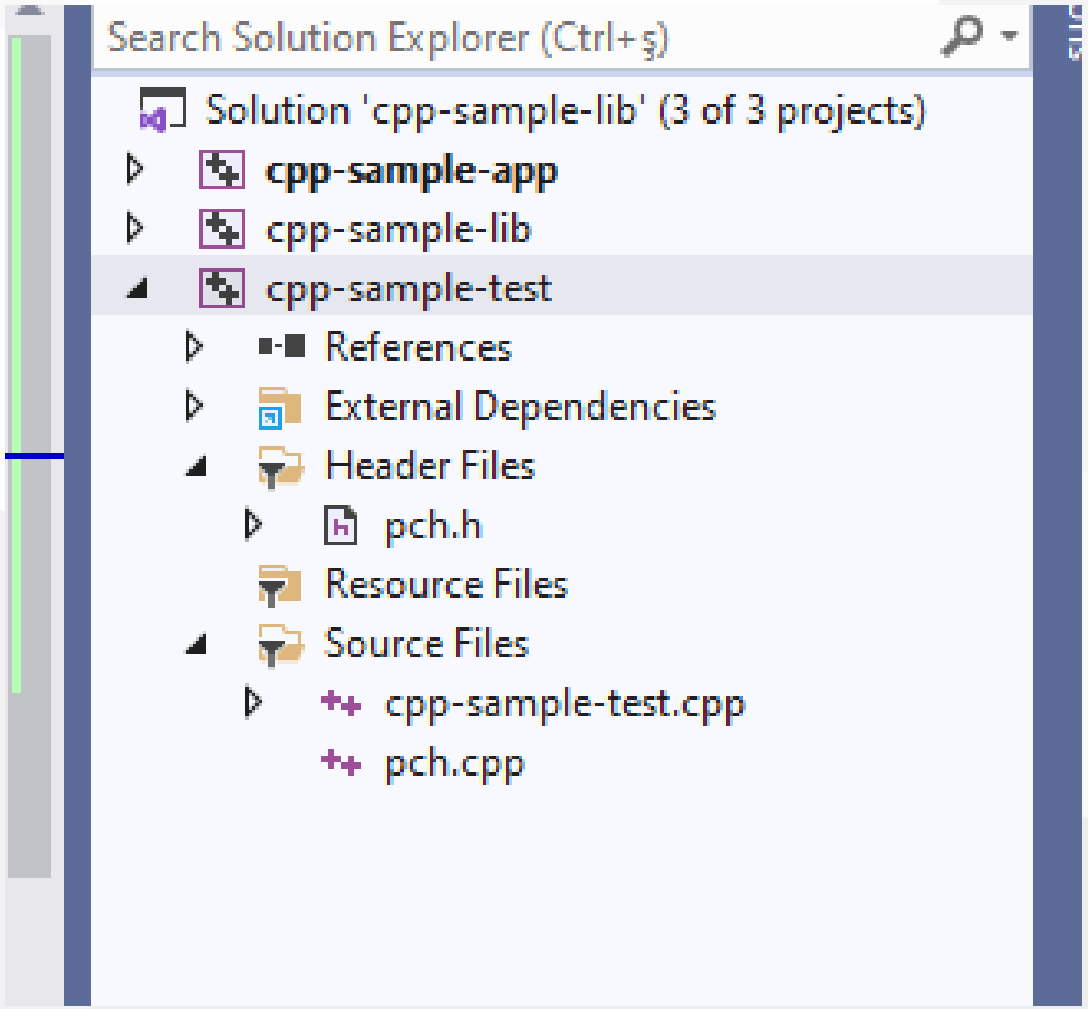
Visual Studio Community Edition - C++ Unit Tests-4

- Set project path and name



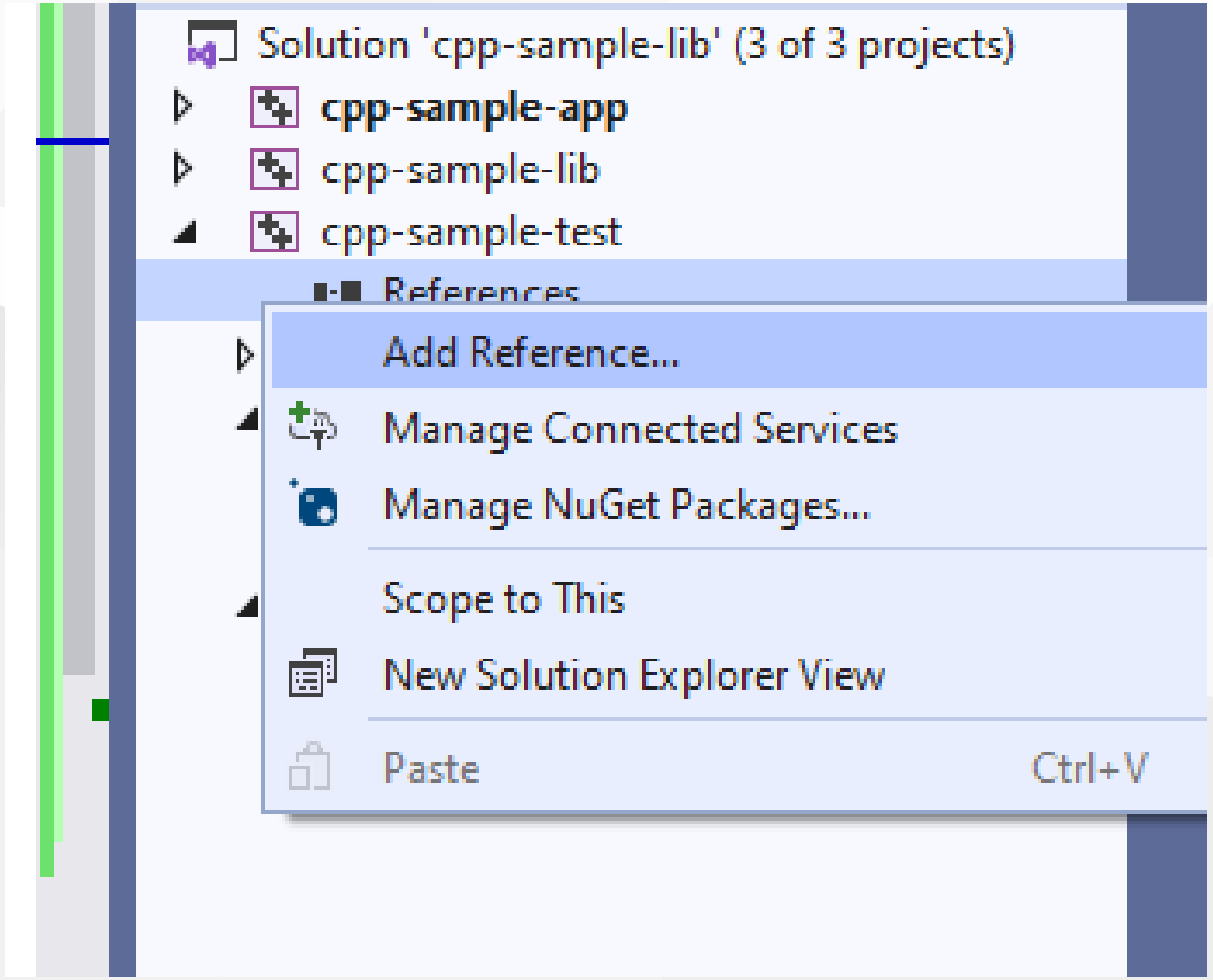
Visual Studio Community Edition - C++ Unit Tests-5

- You will have `cpp-sample-test` project



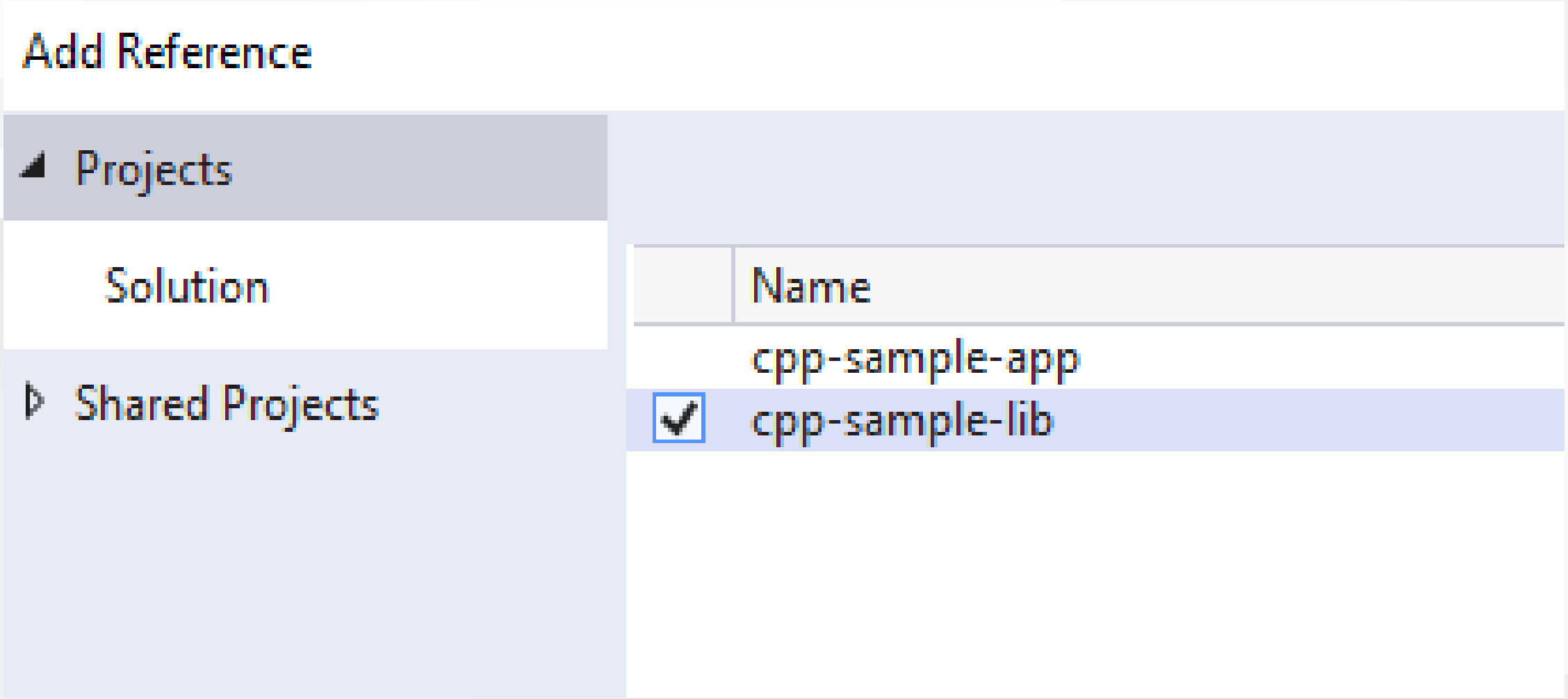
Visual Studio Community Edition - C++ Unit Tests-6

- Add library project from references



Visual Studio Community Edition - C++ Unit Tests-7

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

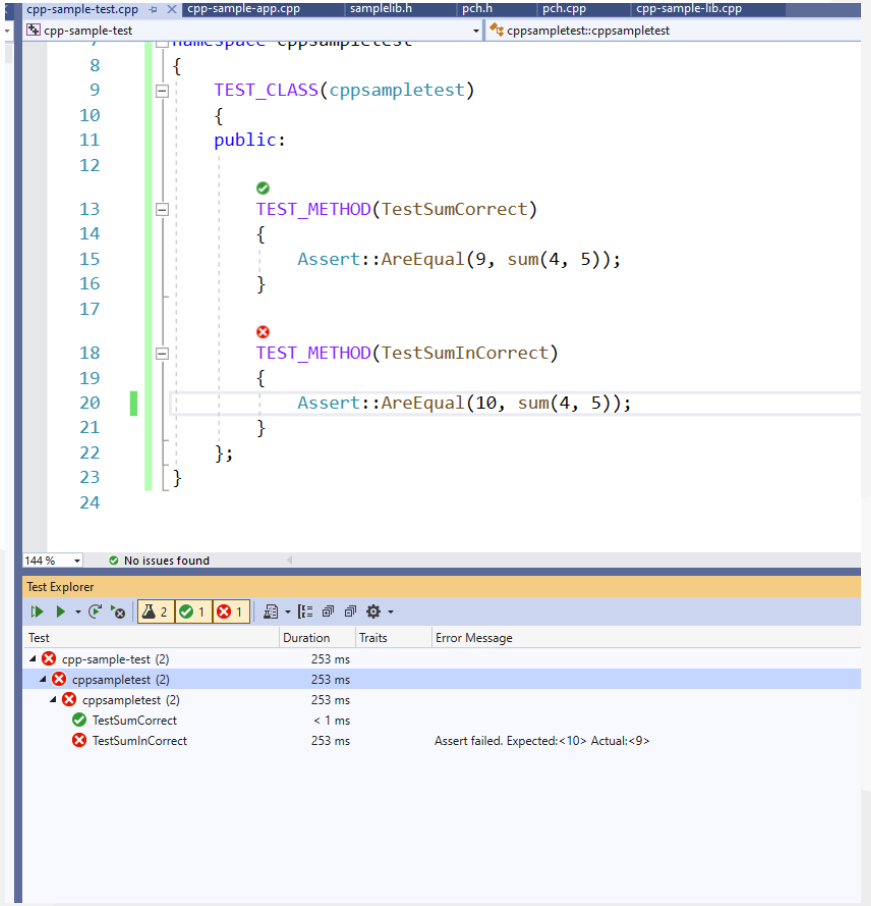


Visual Studio Community Edition - C++ Unit Tests-8

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));
        }
    };
}
```

Visual Studio Community Edition - C++ Unit Tests-9



Visual Studio Community Edition

C# Unit Tests

- MSTest + .Net
- Fine Code Coverage
- NUnit + .NetCore

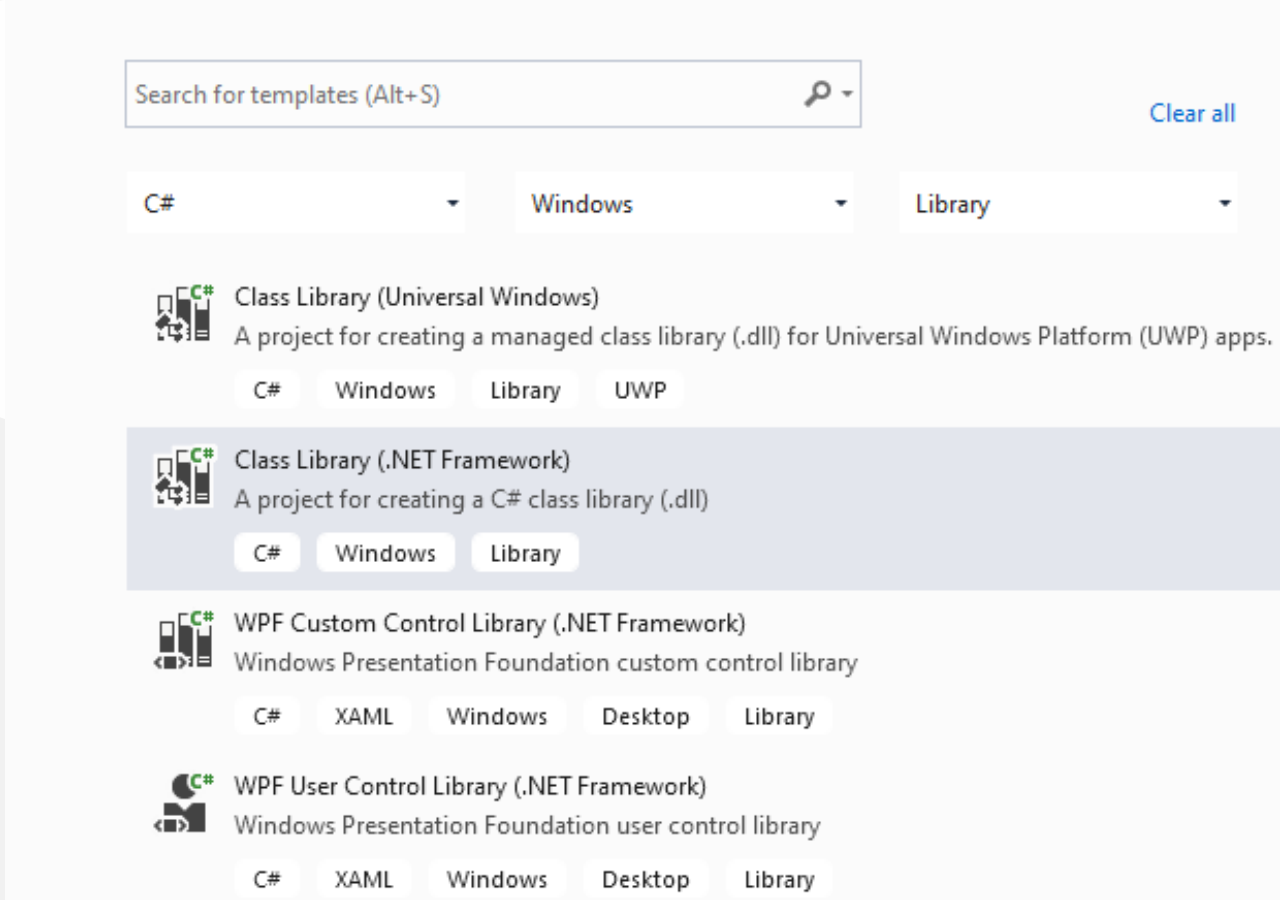
Visual Studio Community Edition (C# Unit Test + MSTestV2+.Net)-1

- Install extension fine code coverage

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

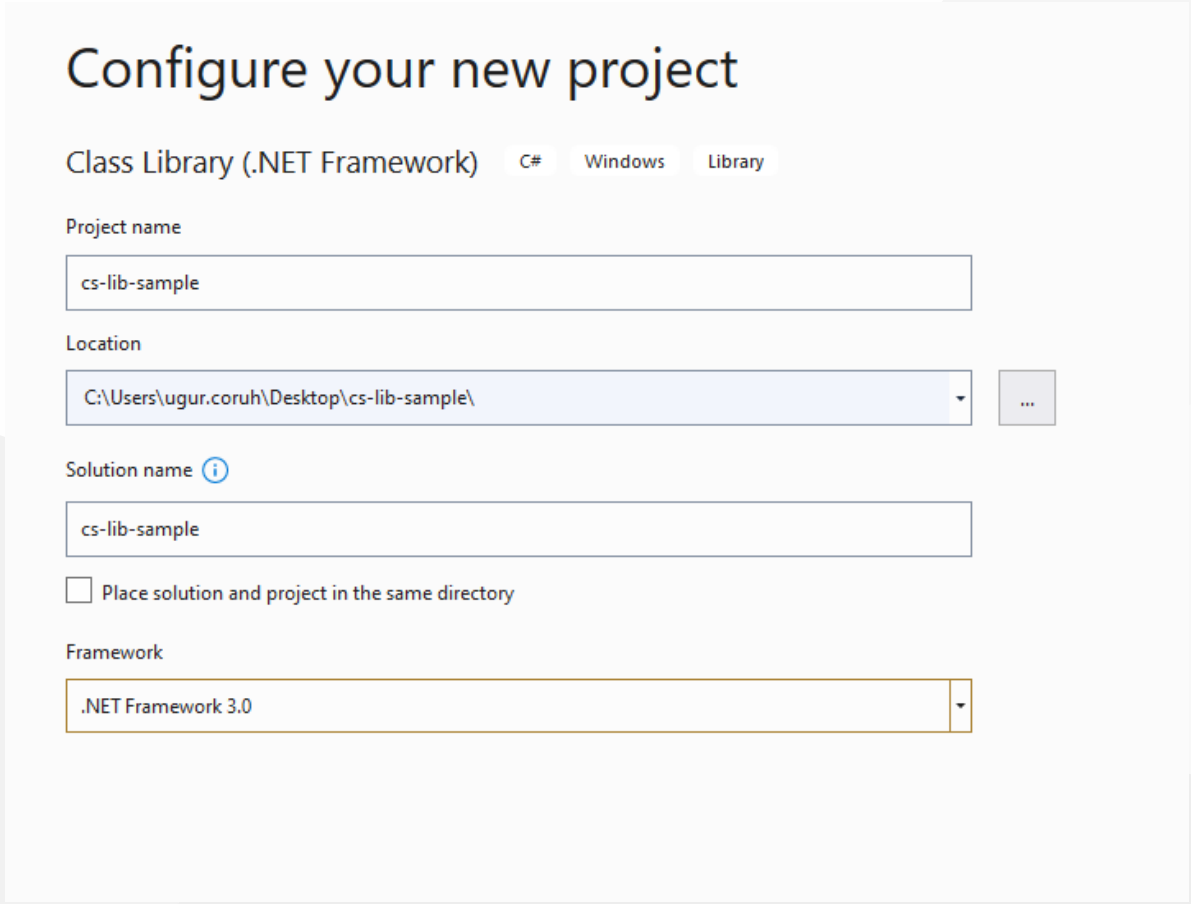
Visual Studio Community Edition (C# Unit Test + MSTestV2+.Net)-2

- Create a .Net Framework Library



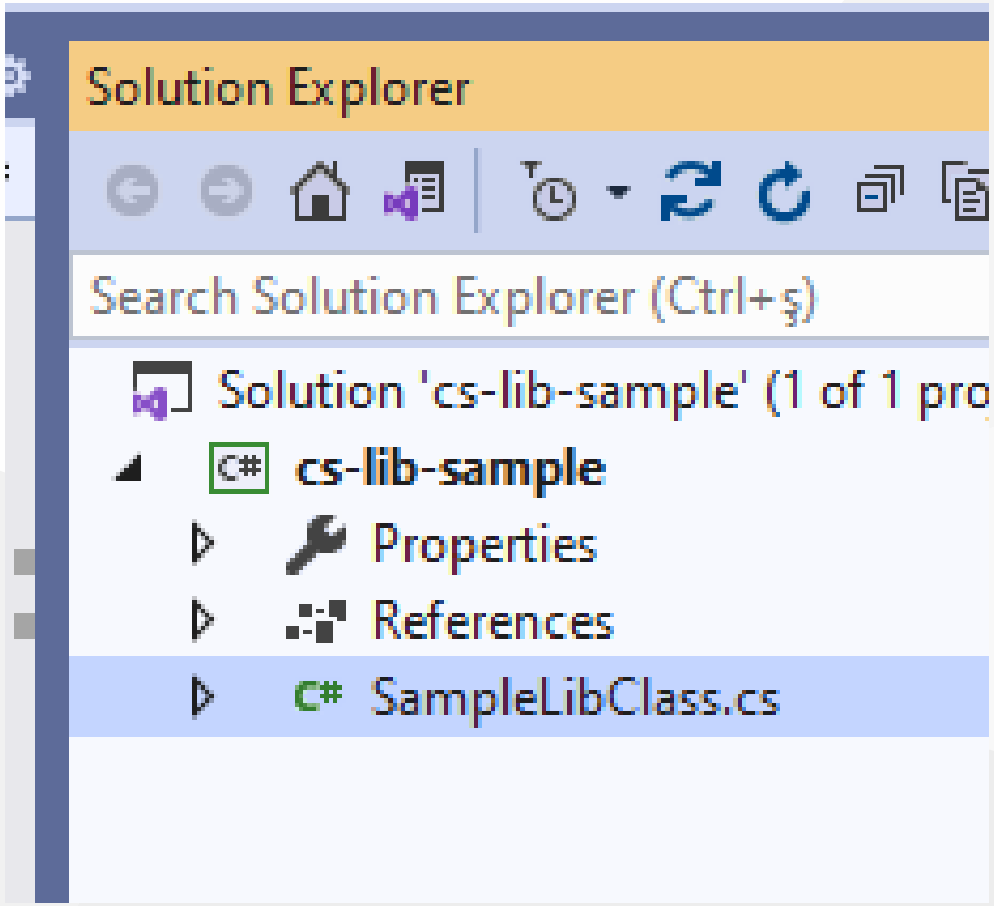
Visual Studio Community Edition (C# Unit Test + MSTestV2+.Net)-3

- Set project framework and path



Visual Studio Community Edition (C# Unit Test + MSTestV2+.Net)-4

- Create library functions

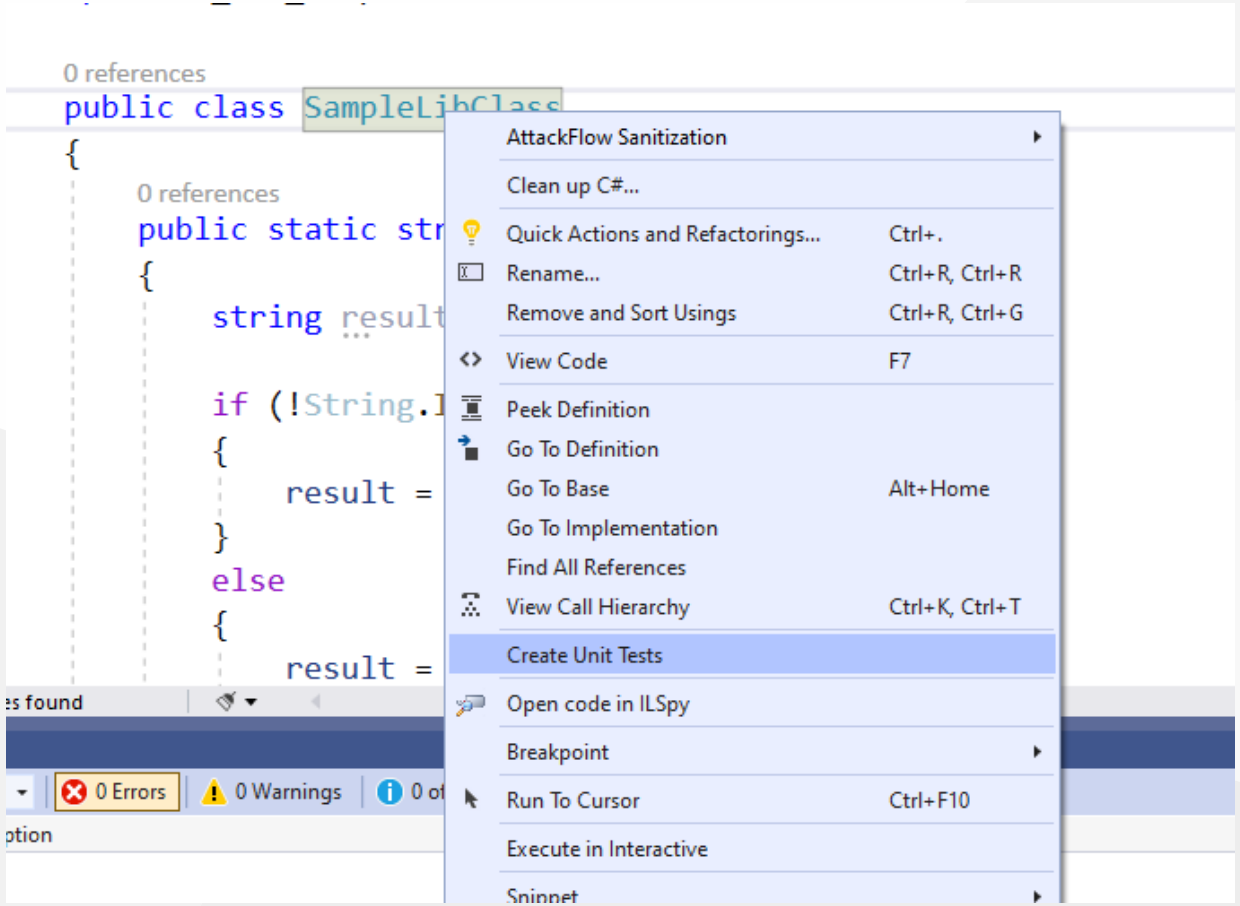


Visual Studio Community Edition (C# Unit Test + MSTestV2+.Net)-5

```
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;
        }
    }
}
```

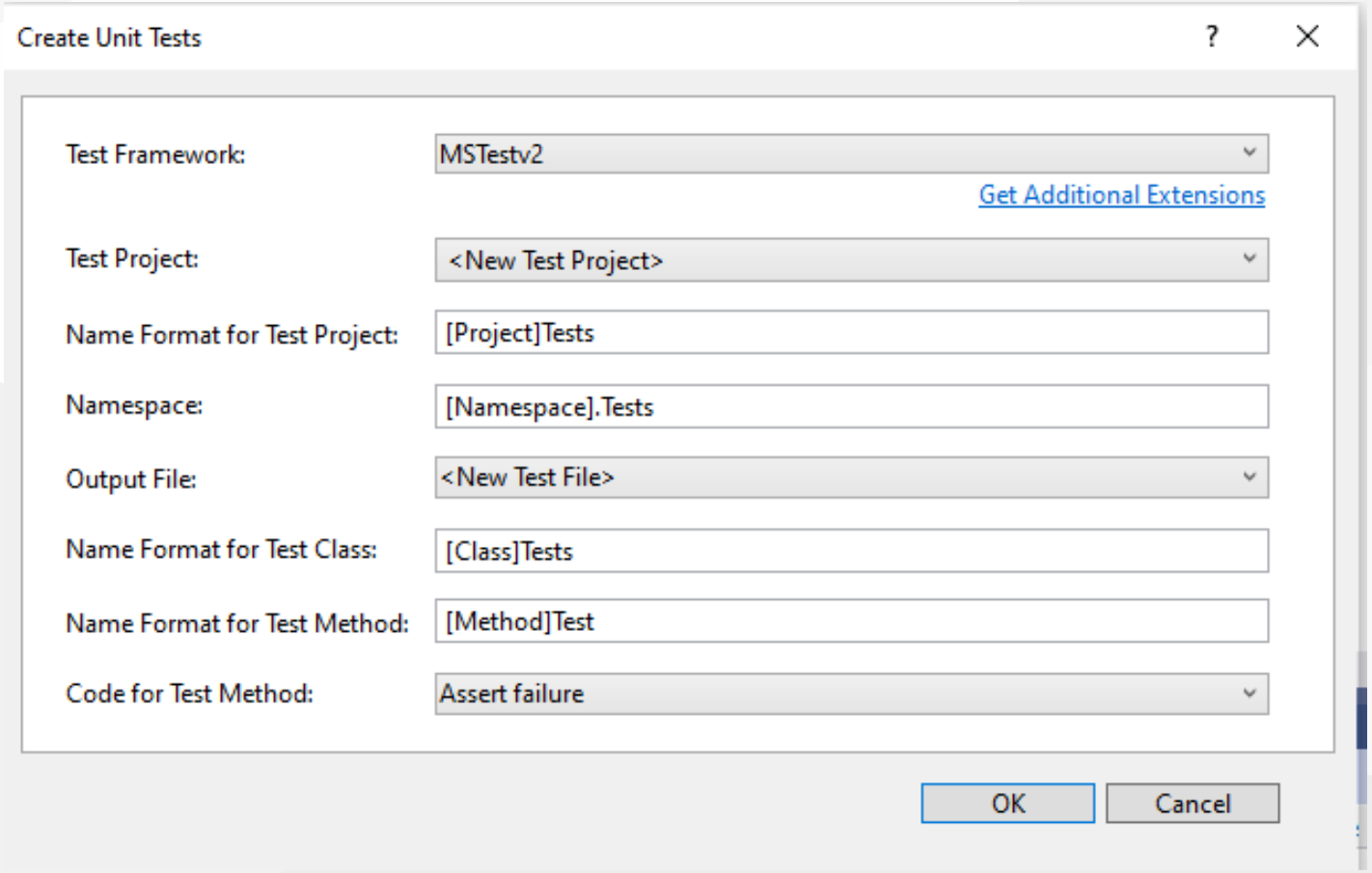
Visual Studio Community Edition (C# Unit Test + MSTestV2+.Net)-6

- Right click and then create unit test project



Visual Studio Community Edition (C# Unit Test + MSTestV2+.Net)-7

- Press OK



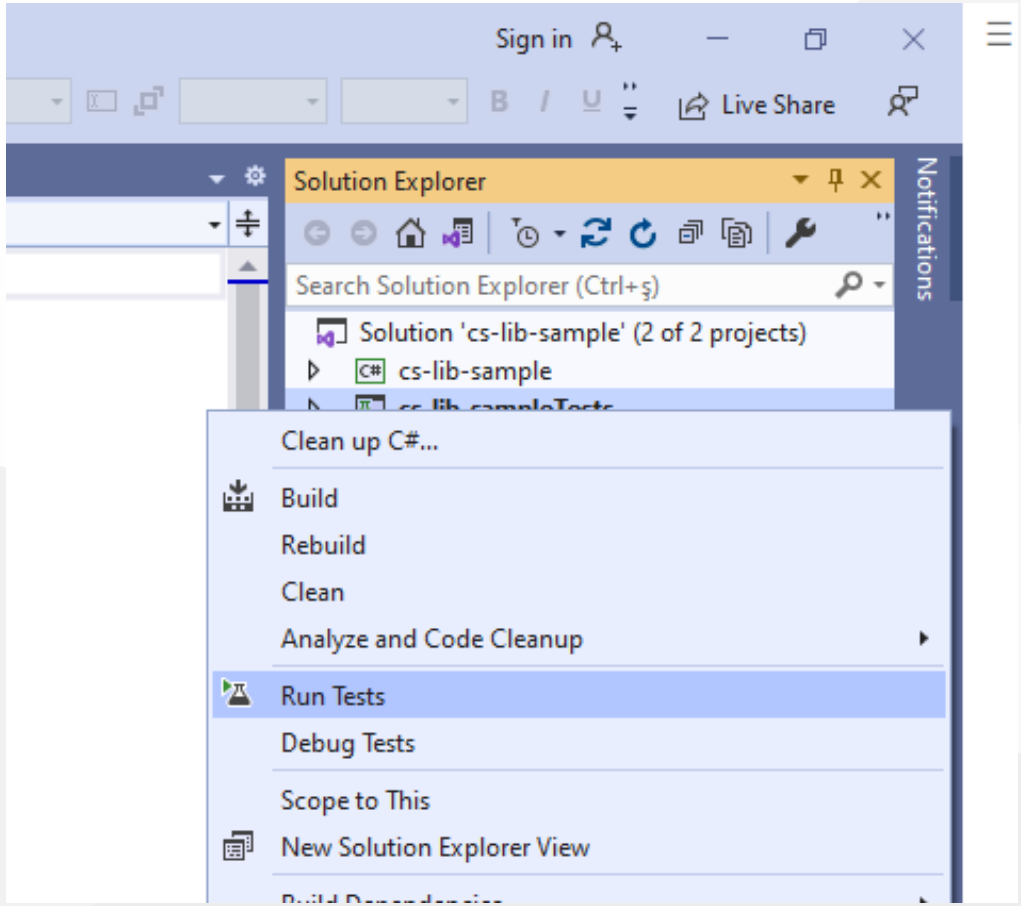
Visual Studio Community Edition (C# Unit Test + MSTestV2+.Net)-8

- 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 should work");
        }
        [TestMethod()]
        public void testSayHelloToWrong()
        {
            Assert.AreEqual("Hello All", SampleLibClass.sayHelloTo("Computer"), "Regular say hello won't work");
        }


        [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");
        }


    }
}
```

Visual Studio Community Edition (C# Unit Test + MSTestV2+.Net)-9

- Run tests



Visual Studio Community Edition (C# Unit Test + MSTestV2+.Net)-10

you will code coverage and entered or passed branches

```
7   public class SampleLibClass
8   {
        2 references | 1/2 passing
9       public static string sayHelloTo(string name)
10      {
11          string result = String.Empty;
12
13          if (!String.IsNullOrEmpty(name))
14          {
15              result = "Hello " + name;
16          }
17          else
18          {
19              result = "Hello There";
20          }
21
22          Console.WriteLine(result);
23
24          return result;
25      }
26
        2 references | 1/2 passing
```

144 % | No issues found | Ln: 18 Ch: 14 SPC CRLF

Fine Code Coverage

Coverage | Summary | Risk Hotspots | Rate & Review | Log Issue/Suggestion | Buy me a coffee

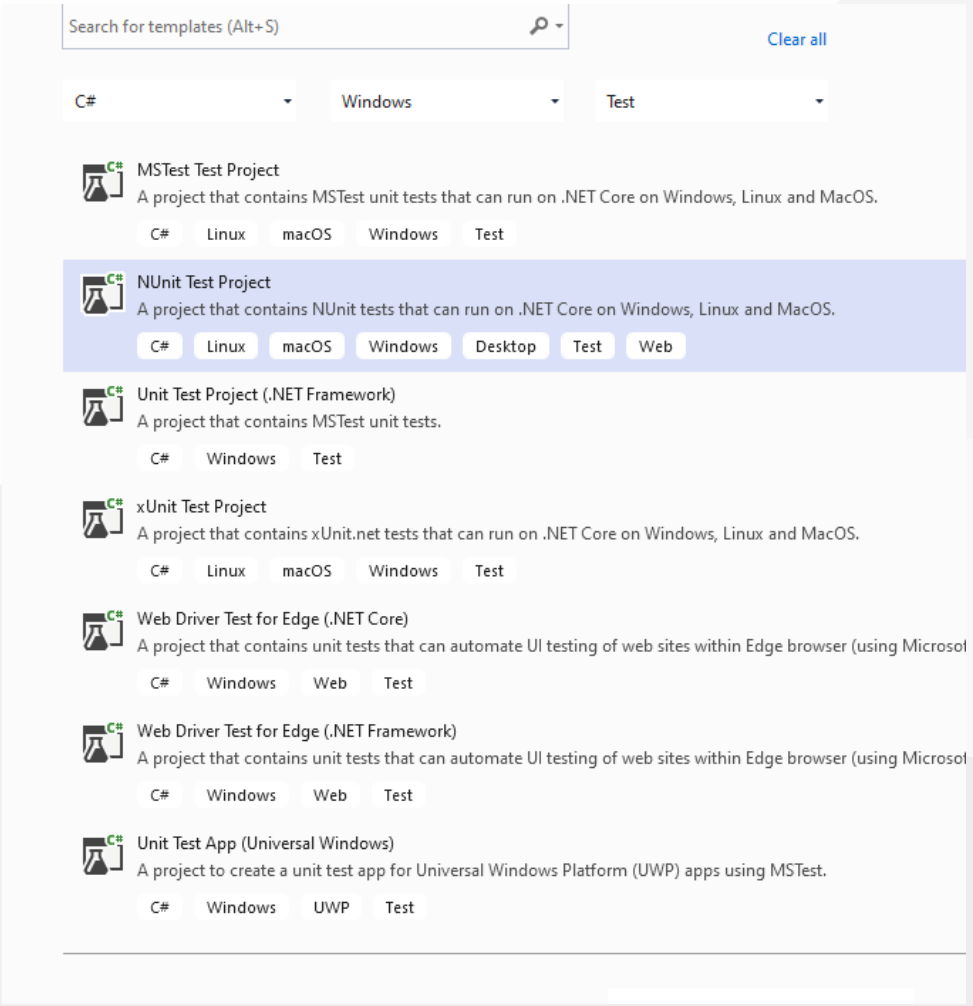
Name	Covered	Uncovered	Coverable	Total	Line coverage
cs-lib-sample	17	3	20	39	85%
SampleLibClass	17	3	20	39	85%
cs-lib-sampleTests	14	2	16	51	87.5%
SampleLibClassTests	14	2	16	51	87.5%

Visual Studio Community Edition

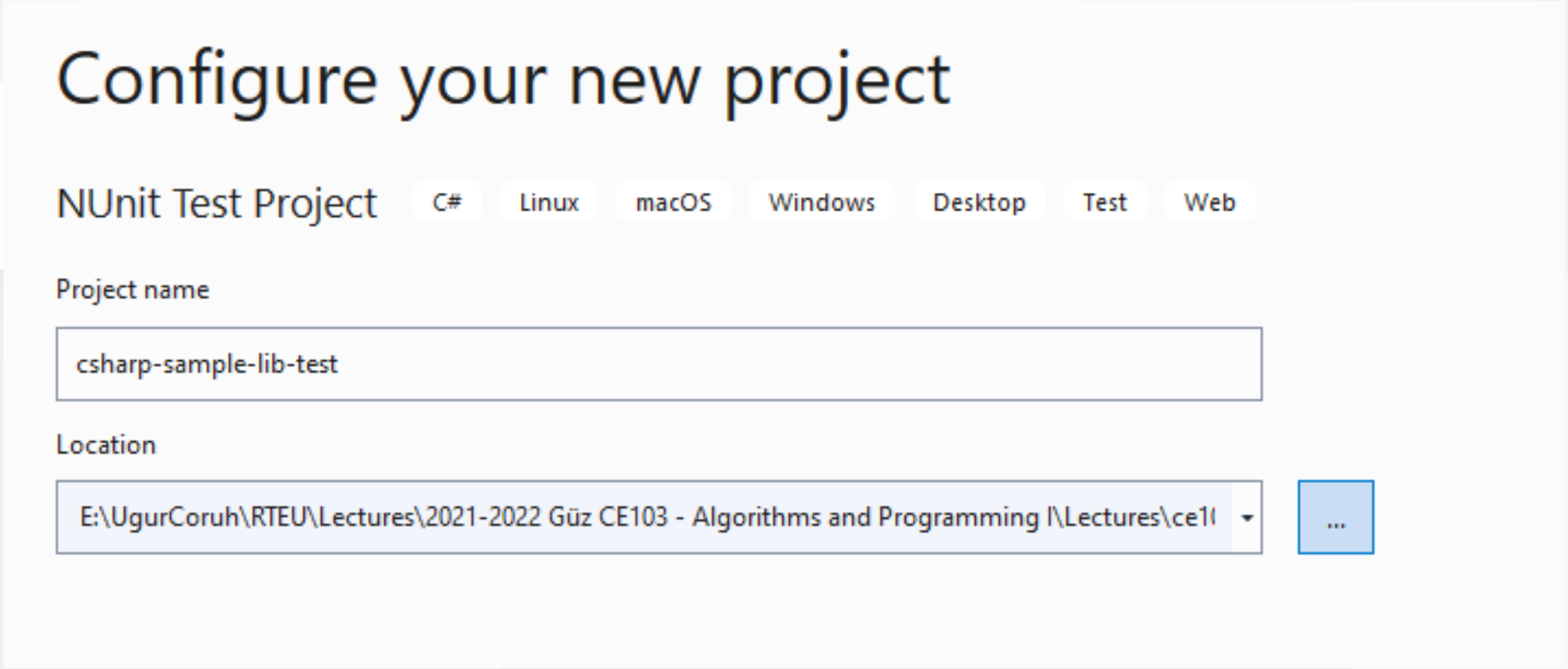
C# Unit Test + NUnit + .NETCore

Visual Studio Community Edition (C# Unit Test+NUnit+.NETCore)-1

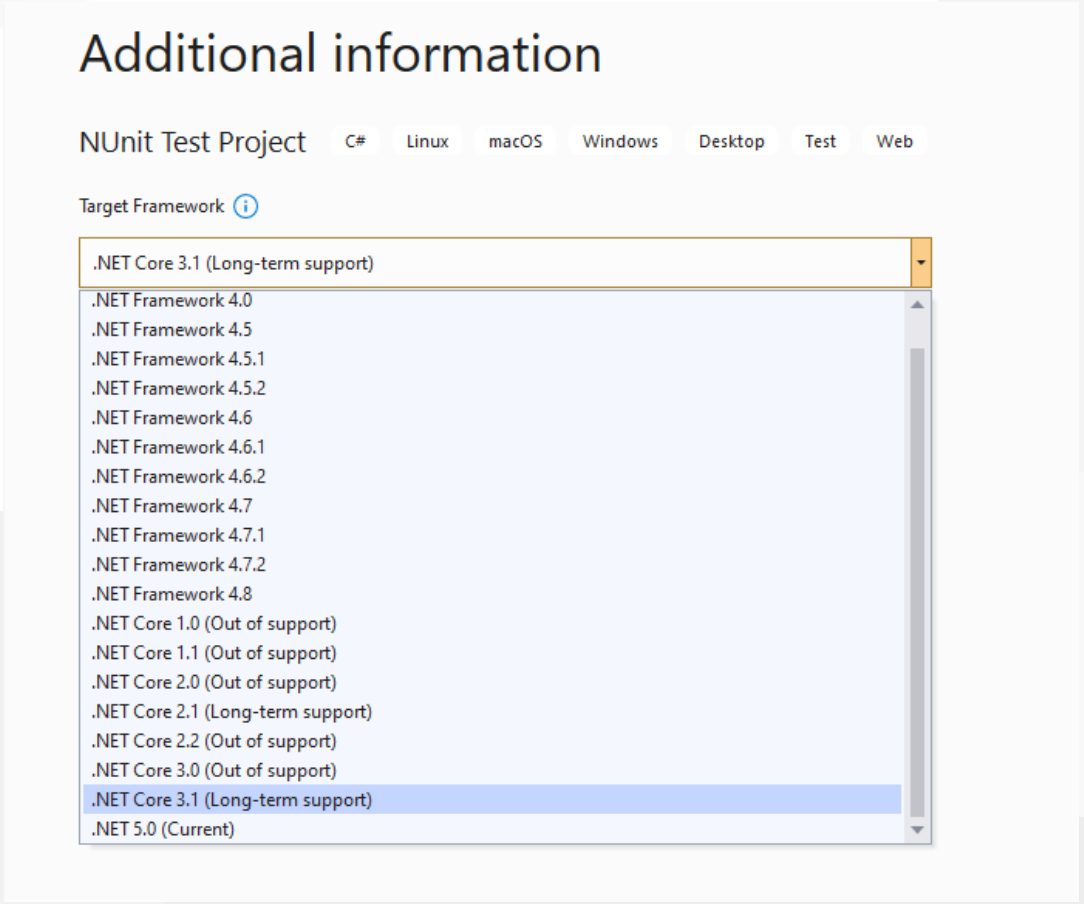
- Use `cshar-sample-lib` for this example
- Create and add a unit test project to solution



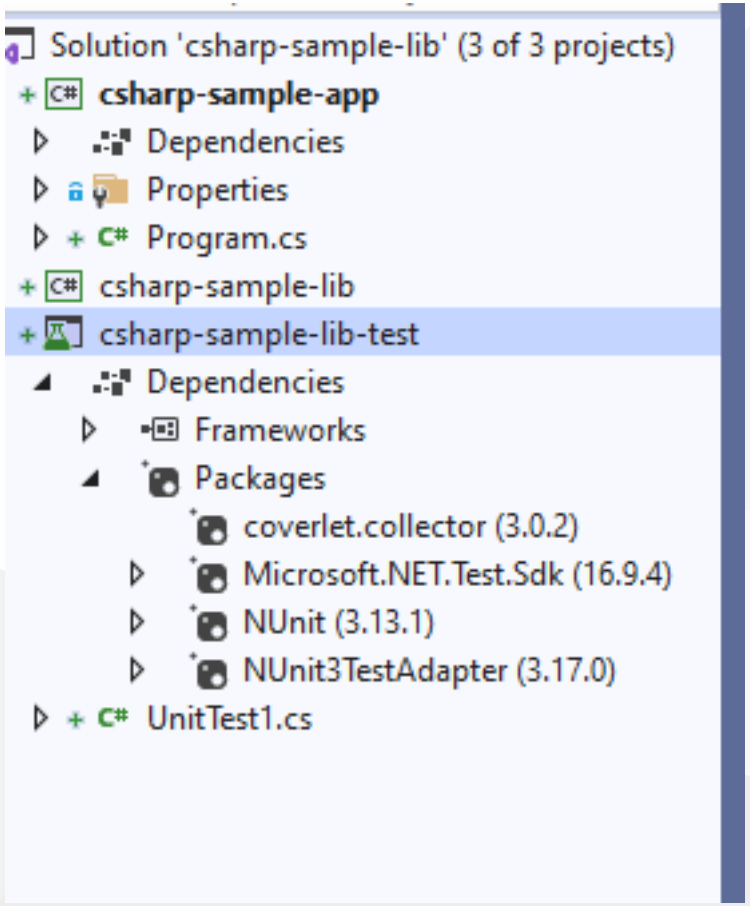
Visual Studio Community Edition (C# Unit Test+NUnit+.NETCore)-2



Visual Studio Community Edition (C# Unit Test+NUnit+.NETCore)-3

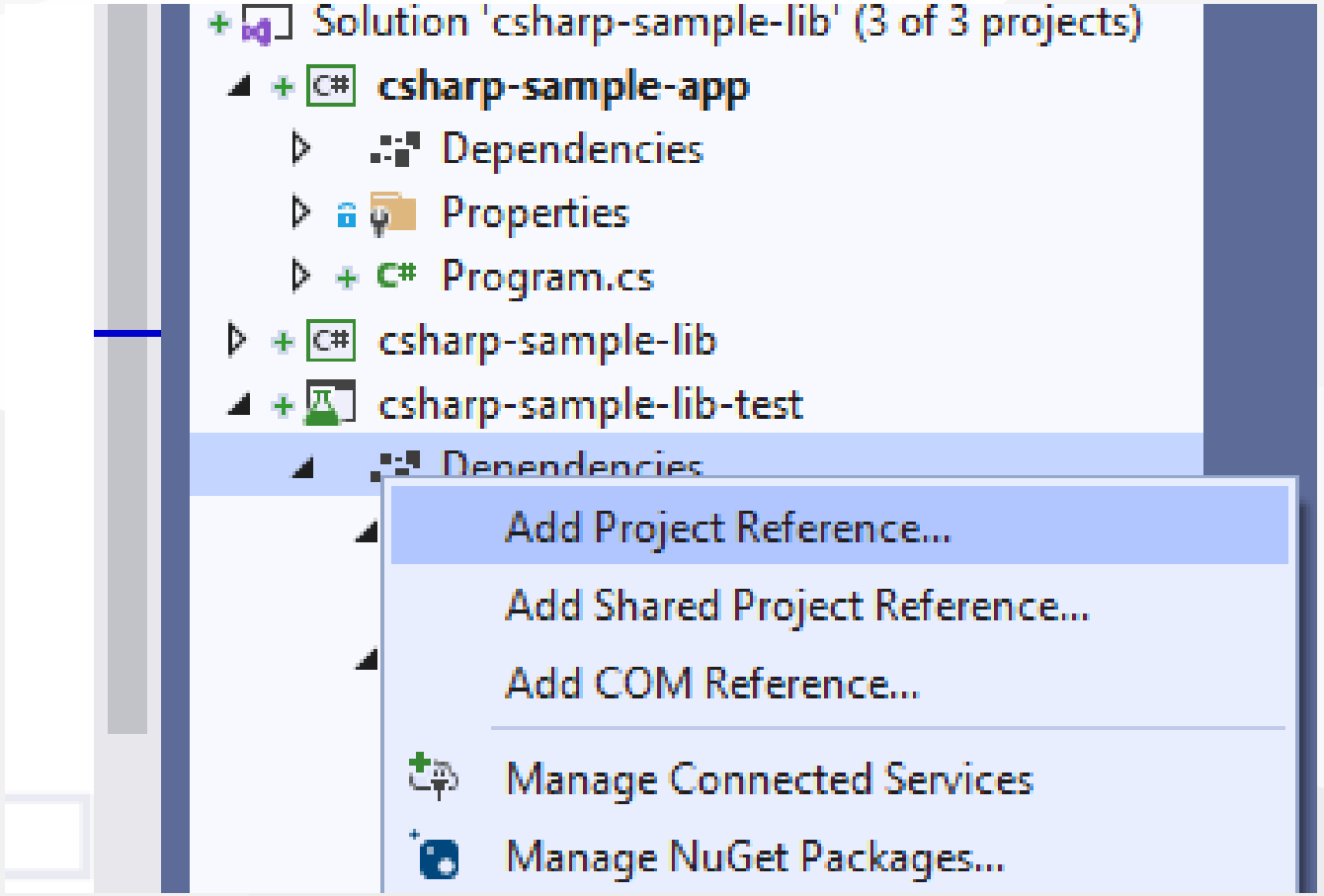


Visual Studio Community Edition (C# Unit Test+NUnit+.NETCore)-4

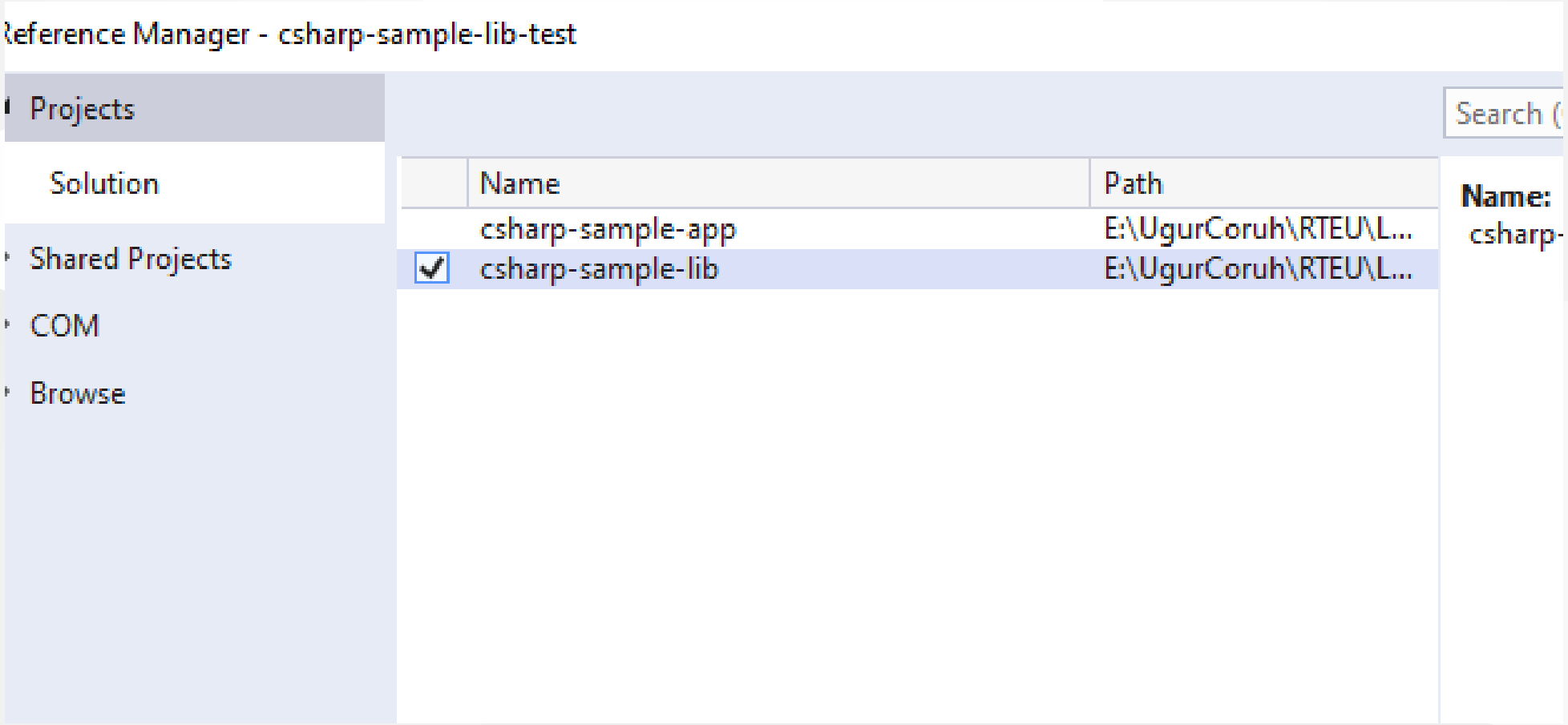


Visual Studio Community Edition (C# Unit Test+NUnit+.NETCore)-5

- Add project reference



Visual Studio Community Edition (C# Unit Test+NUnit+.NETCore)-6



Visual Studio Community Edition (C# Unit Test+NUnit+.NETCore)-7

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 should work");
        }

        [Test]
        public void testSayHelloToWrong()
        {
            Assert.AreEqual("Hello All", sampleLibClass.sayHelloTo("Computer"), "Regular say hello won't work");
        }


        [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");
        }
    }
}
```

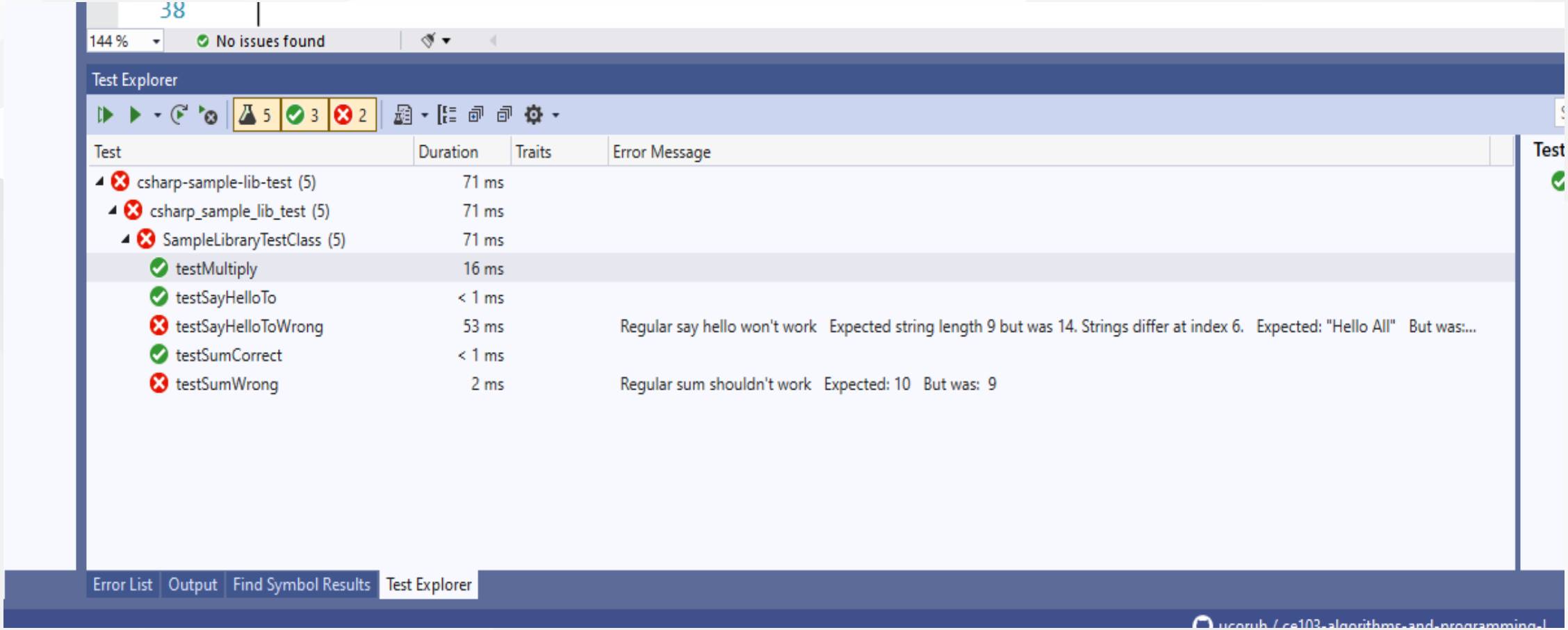
Visual Studio Community Edition (C# Unit Test+NUnit+.NETCore)-8

- 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;
        }
    }
}
```

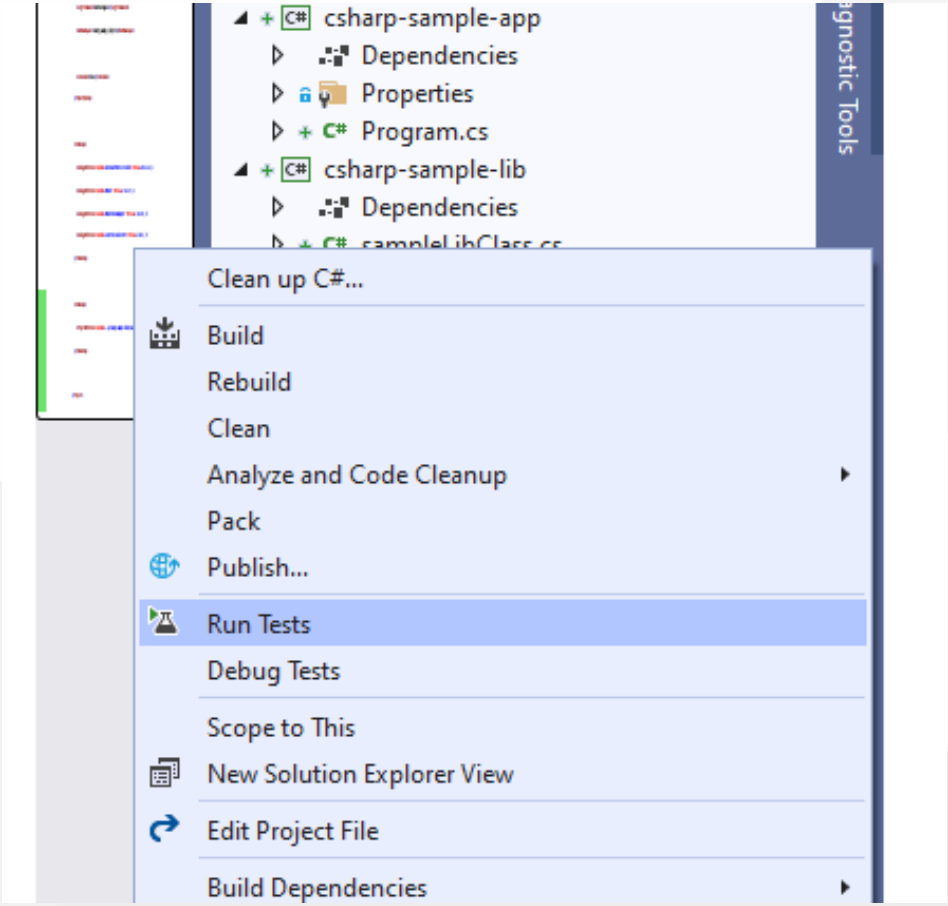
Visual Studio Community Edition (C# Unit Test+NUnit+.NETCore)-9

- Open test explorer and run tests



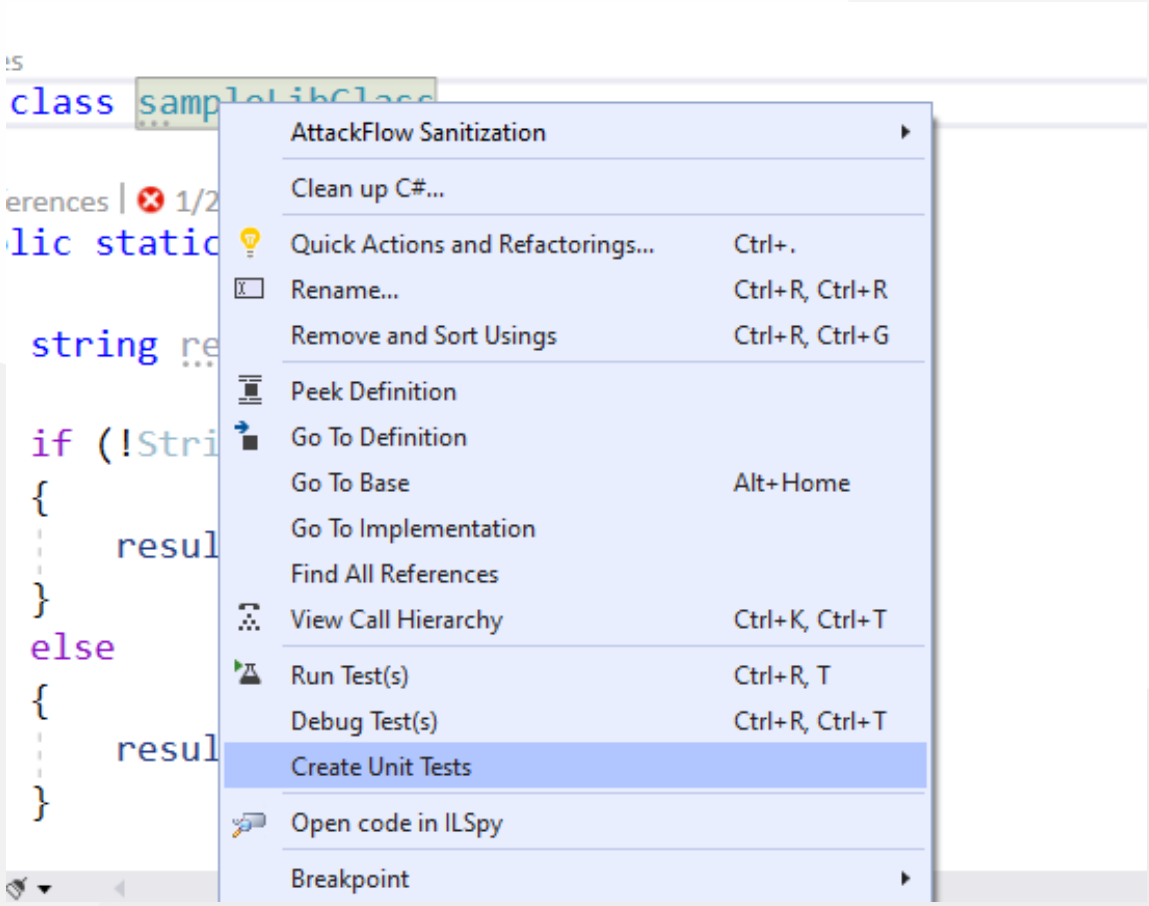
Visual Studio Community Edition (C# Unit Test+NUnit+.NETCore)-10

- or you can run from project

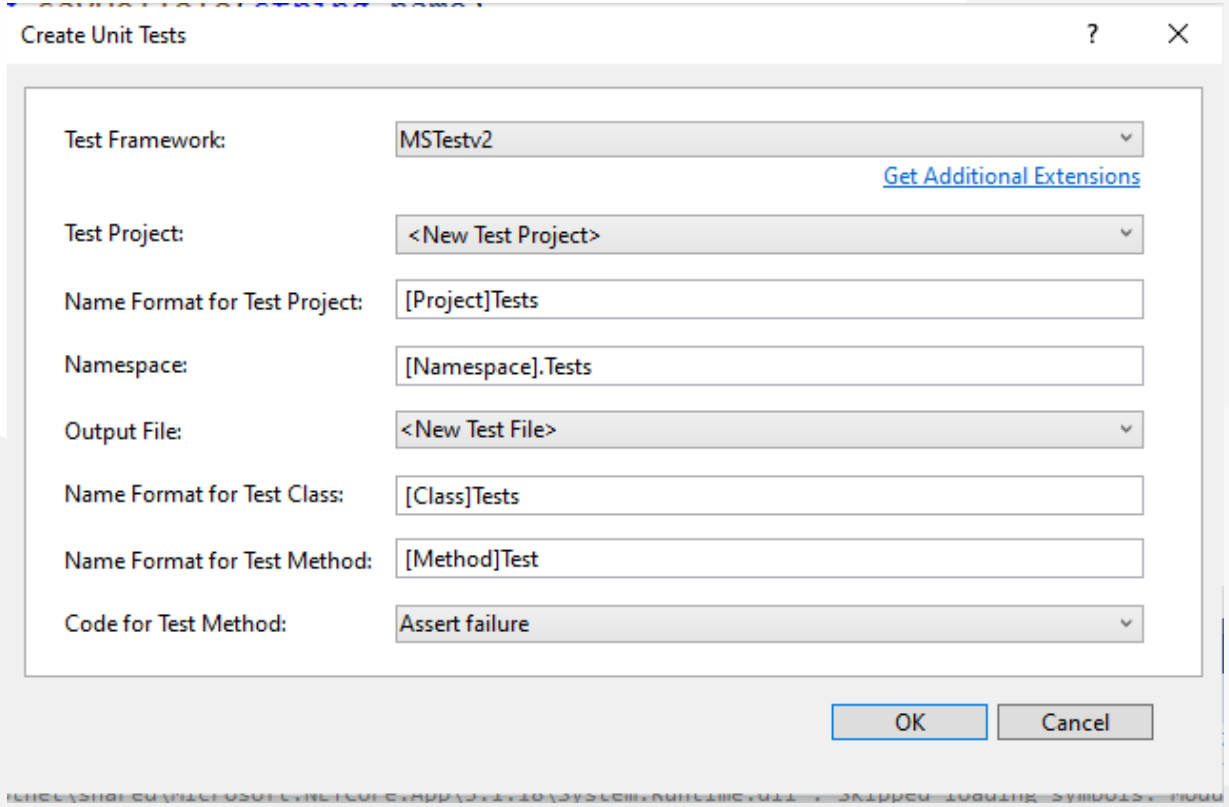


Visual Studio Community Edition (C# Unit Test+NUnit+.NETCore)-11

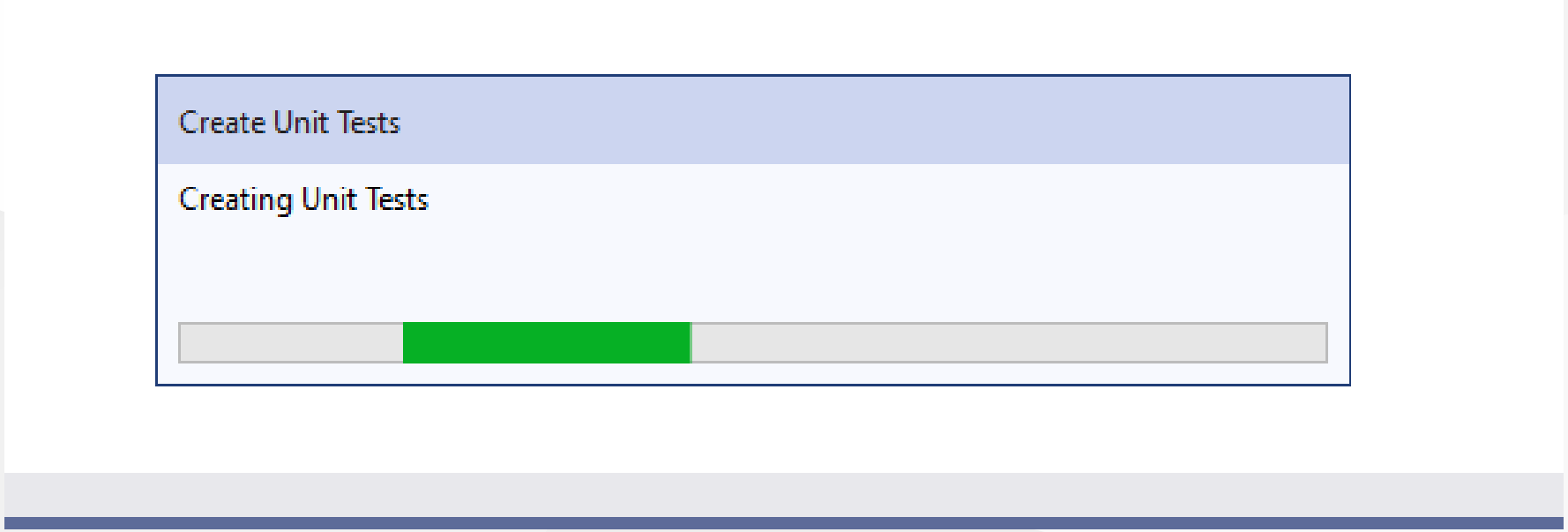
- Also we can create unit test from library class,
- Right click the sampleLibClass and select create unit tests but this option do not provide nunit tests.



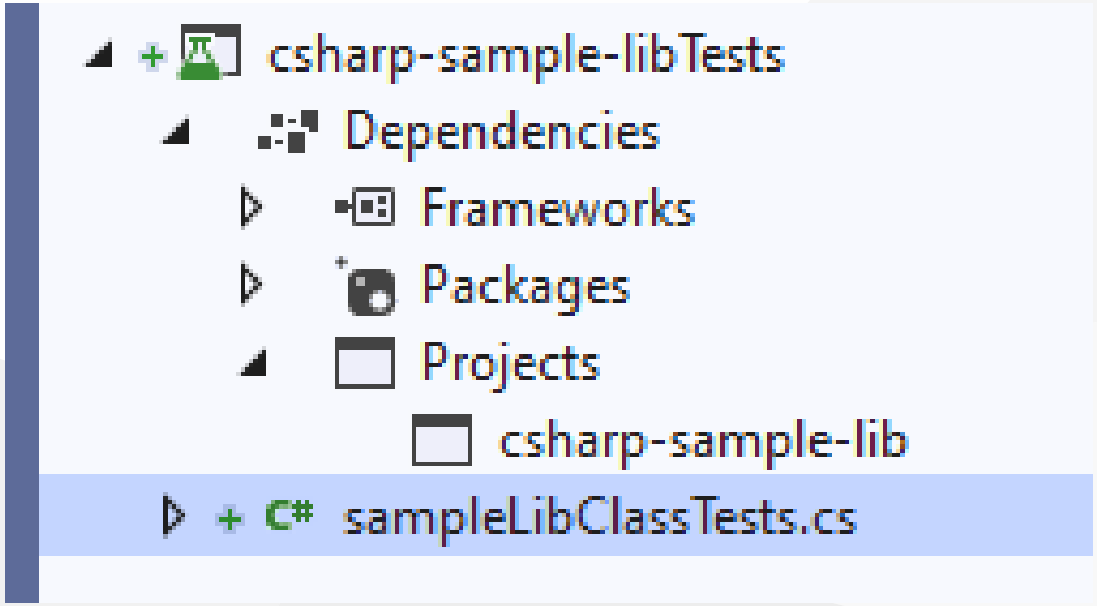
Visual Studio Community Edition (C# Unit Test+NUnit+.NETCore)-12



Visual Studio Community Edition (C# Unit Test+NUnit+.NETCore)-13



Visual Studio Community Edition (C# Unit Test+NUnit+.NETCore)-14

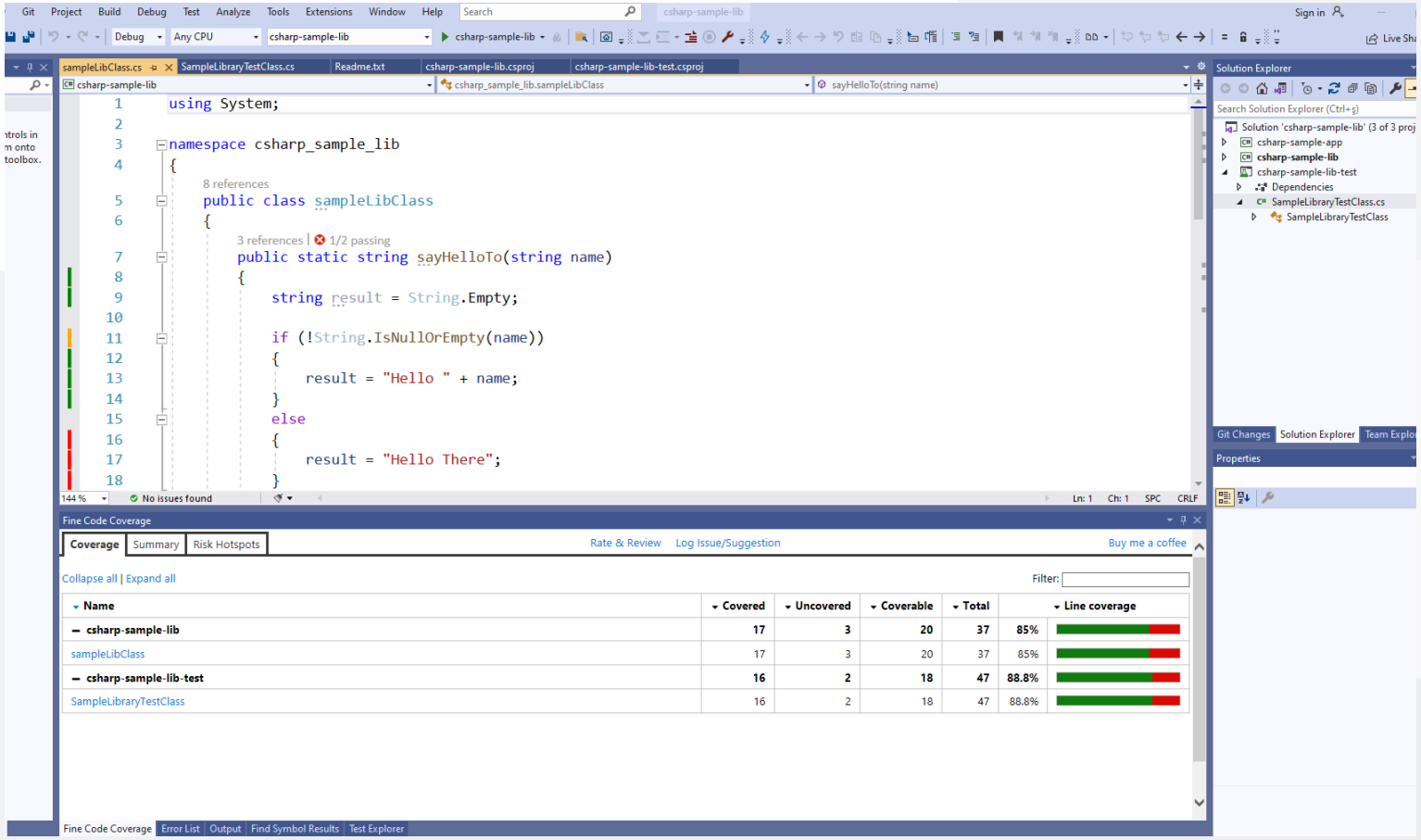


Visual Studio Community Edition (C# Unit Test+NUnit+.NETCore)-15

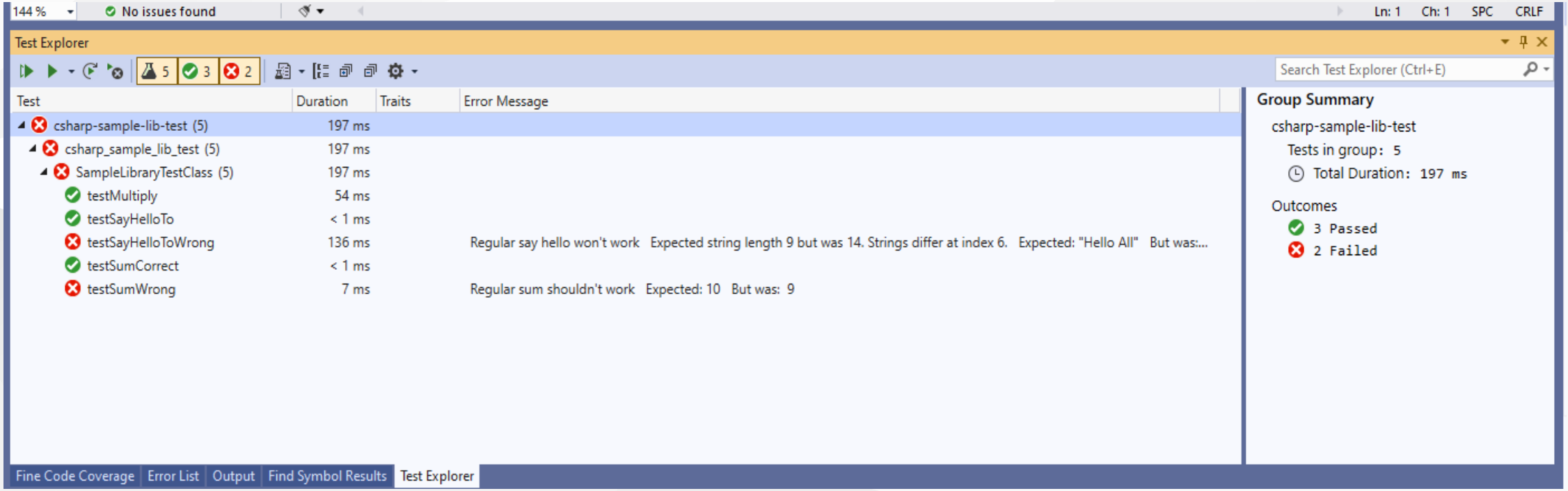
```
using Microsoft.VisualStudio.TestTools.UnitTesting;
using csharp_sample_lib;
using System;
using System.Collections.Generic;
using System.Text;

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();
        }
    }
}
```

Visual Studio Community Edition (C# Unit Test+NUnit+.NETCore)-16

- 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



Visual Studio Community Edition (C# Unit Test+NUnit+.NETCore)-17



- Inline code highlight is part of enterprise visual studio edition
  - Analyzing code coverage in Visual Studio - DEV Community

Visual Studio Community Edition (C# Unit Test+OpenCover + Nunit Runner + Report)

Visual Studio Community Edition (C# Unit Test+OpenCover + Nunit Runner + Report)-1

TL;DR

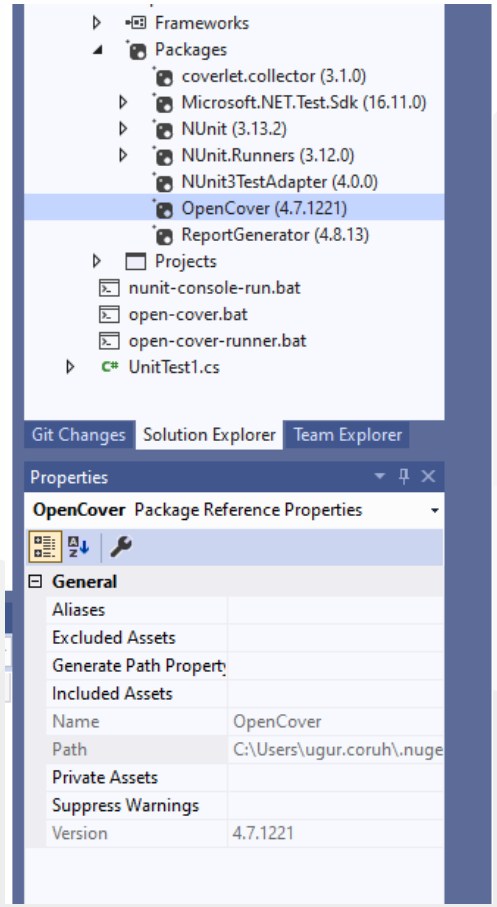
- Additional information you can use OpenCover + Nunit Runner + Report Generator together to setup a code coverage report but it has complex batch running process. After a few try I decided to use fine code coverage but here is the usage not tested well.
- First unit test runner tool doesn't support .Net Core

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

- Follow the instructions on the link
    - CMD OpenCover · sukhoi1/Useful-Notes Wiki · GitHub
- Install OpenCover, ReportGenerator, Nunit,Runners packages then use the package installation folder to get tools that you need

Visual Studio Community Edition (C# Unit Test+OpenCover + Nunit Runner + Report)-2

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



Visual Studio Community Edition (C# Unit Test+OpenCover + Nunit Runner + Report)-3

- 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\netcoreapp3.1

"%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" -targetdir:""%dllpath%\coverage"
pause
```

Visual Studio Community Edition (C# Unit Test+OpenCover + Nunit Runner + Report)-4

- but `nunit3-console.exe` gives error

```
   at NUnit.Engine.Runners.ProcessRunner.RunTests(ITestEventListener listener, TestFilter filter)
--
InvalidCastException
Unable to cast transparent proxy to type 'System.Web.UI.ICallbackEventHandler'.
   at NUnit.Framework.Api.FrameworkController.LoadTestsAction..ctor(FrameworkController controller, Object handler)

Test Run Summary
  Overall result: Failed
  Test Count: 0, Passed: 0, Failed: 0, Warnings: 0, Inconclusive: 0, Skipped: 0
  Start time: 2021-10-24 23:01:09Z
```

Visual Studio Community Edition (C# Unit Test+OpenCover + Nunit Runner + Report)-5

- 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
  - Code coverage of .NET Core unit tests with OpenCover – Automation Rhapsody
- Sample OpenCover report
  - Summary - Coverage Report

Visual Studio Community Edition (C# Unit Test+OpenCover + Nunit Runner + Report)-6

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

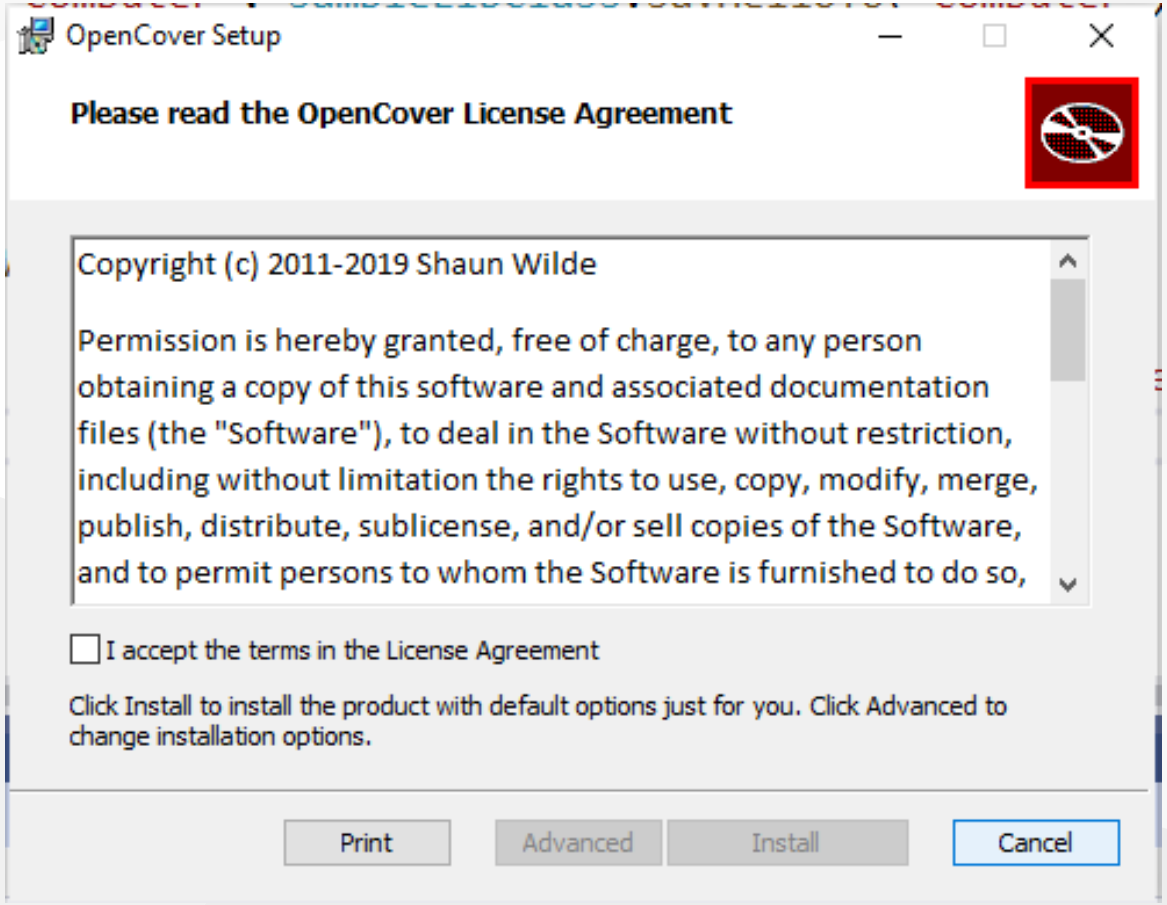
- You can also download the tools from github project pages and install on your operating system,

Visual Studio Community Edition (C# Unit Test+OpenCover + Nunit Runner + Report)-7

OpenCover

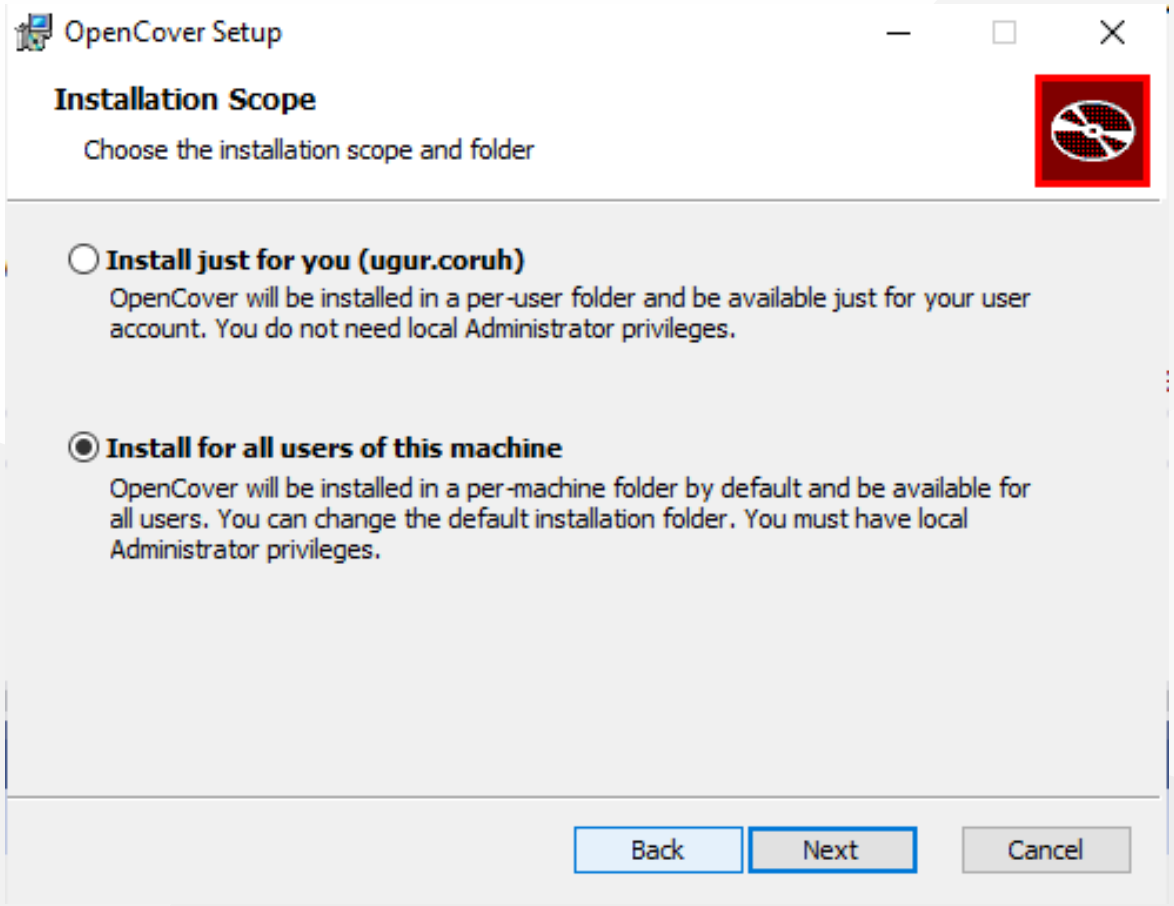
- Releases · OpenCover/opencover · GitHub

Visual Studio Community Edition (C# Unit Test+OpenCover + Nunit Runner + Report)-8

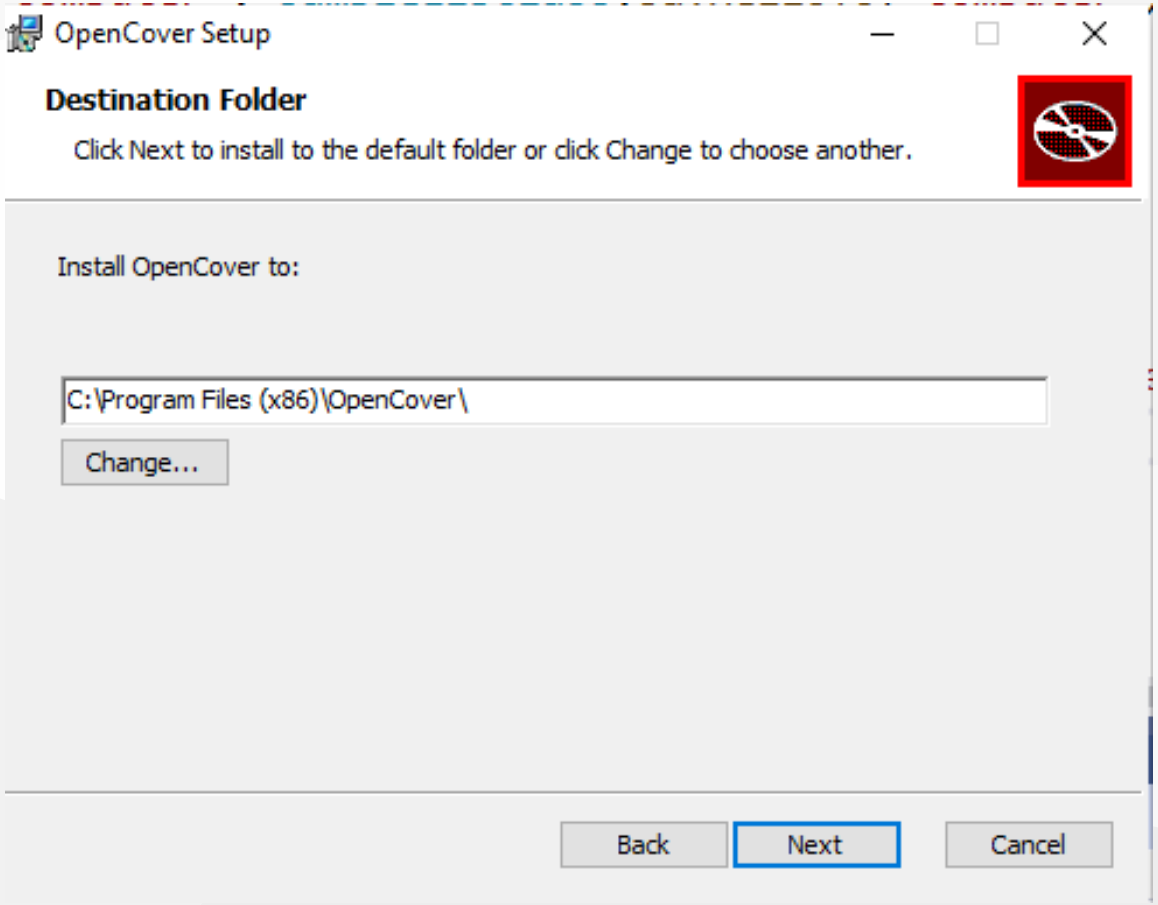


Visual Studio Community Edition (C# Unit Test+OpenCover + Nunit Runner + Report)-9

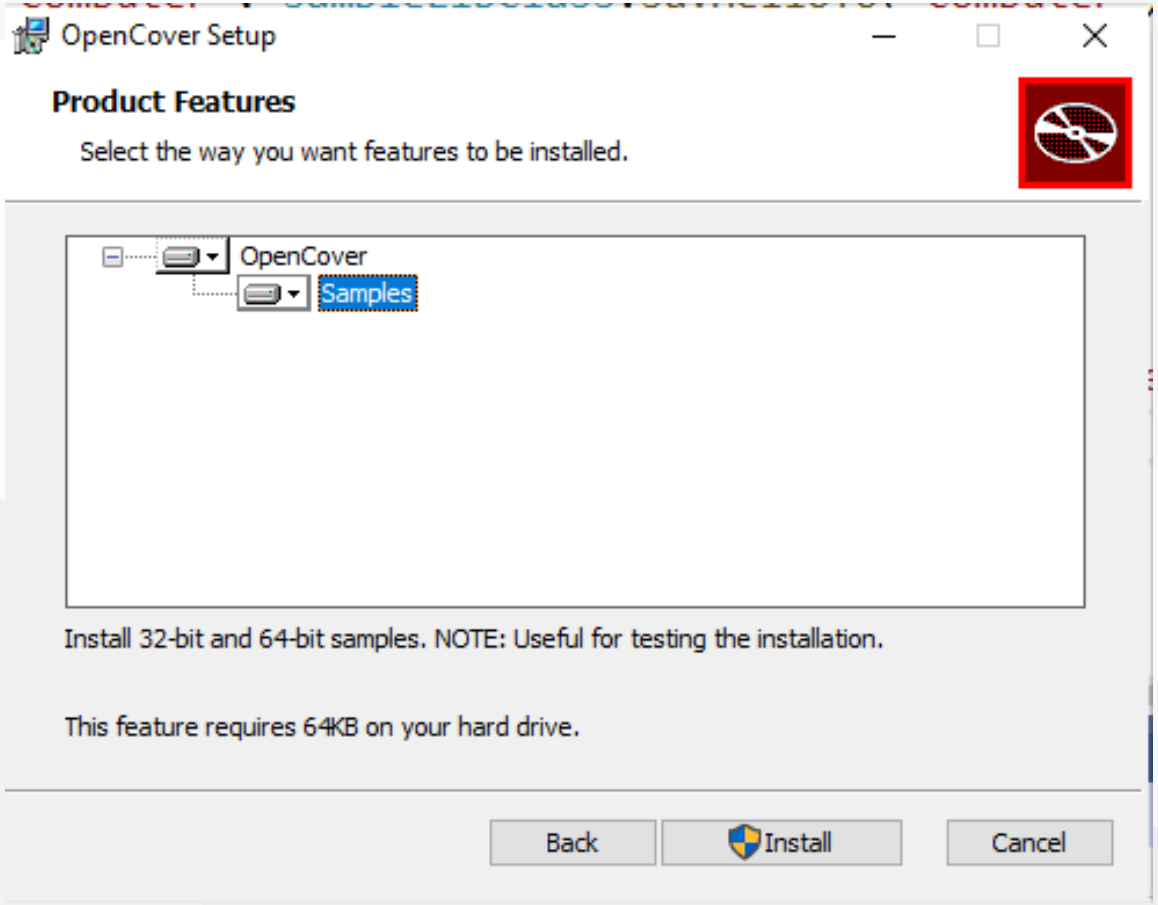
Select advanced and then install for all users



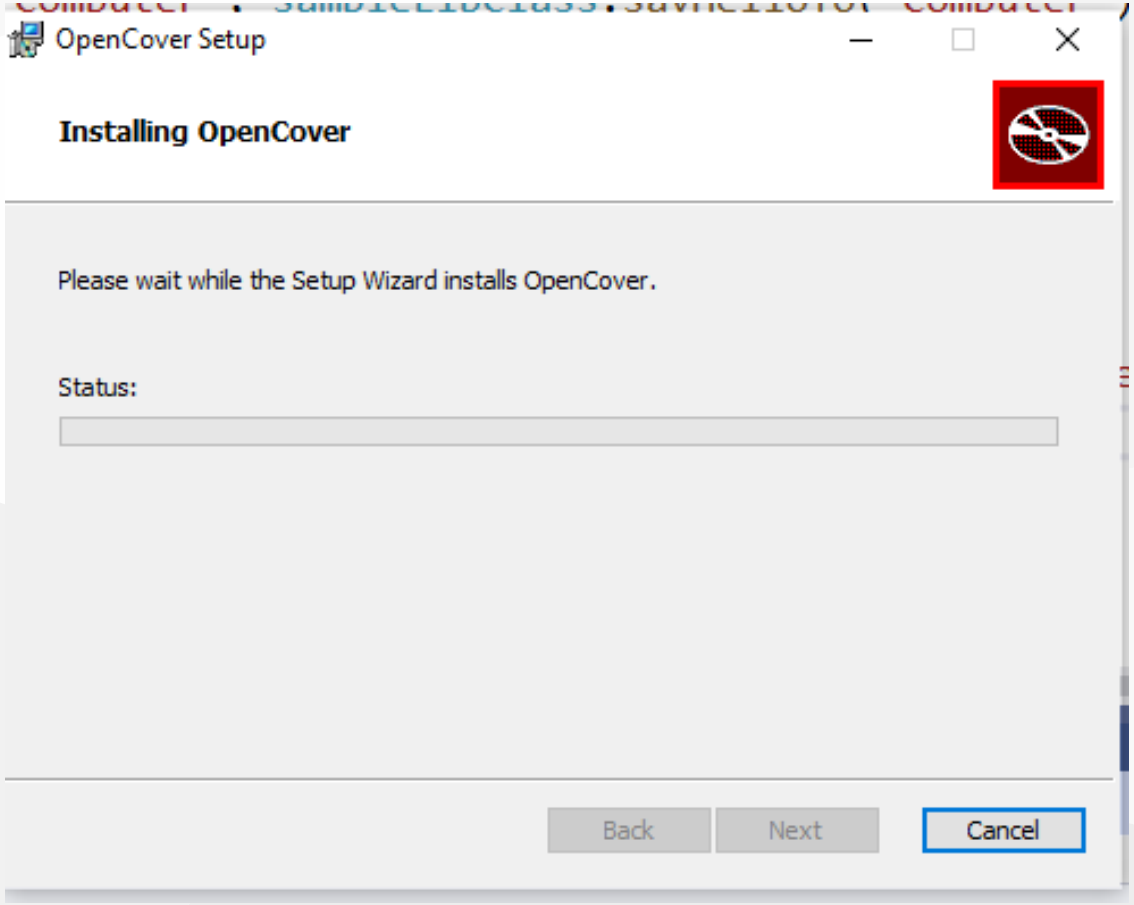
Visual Studio Community Edition (C# Unit Test+OpenCover + Nunit Runner + Report)-10



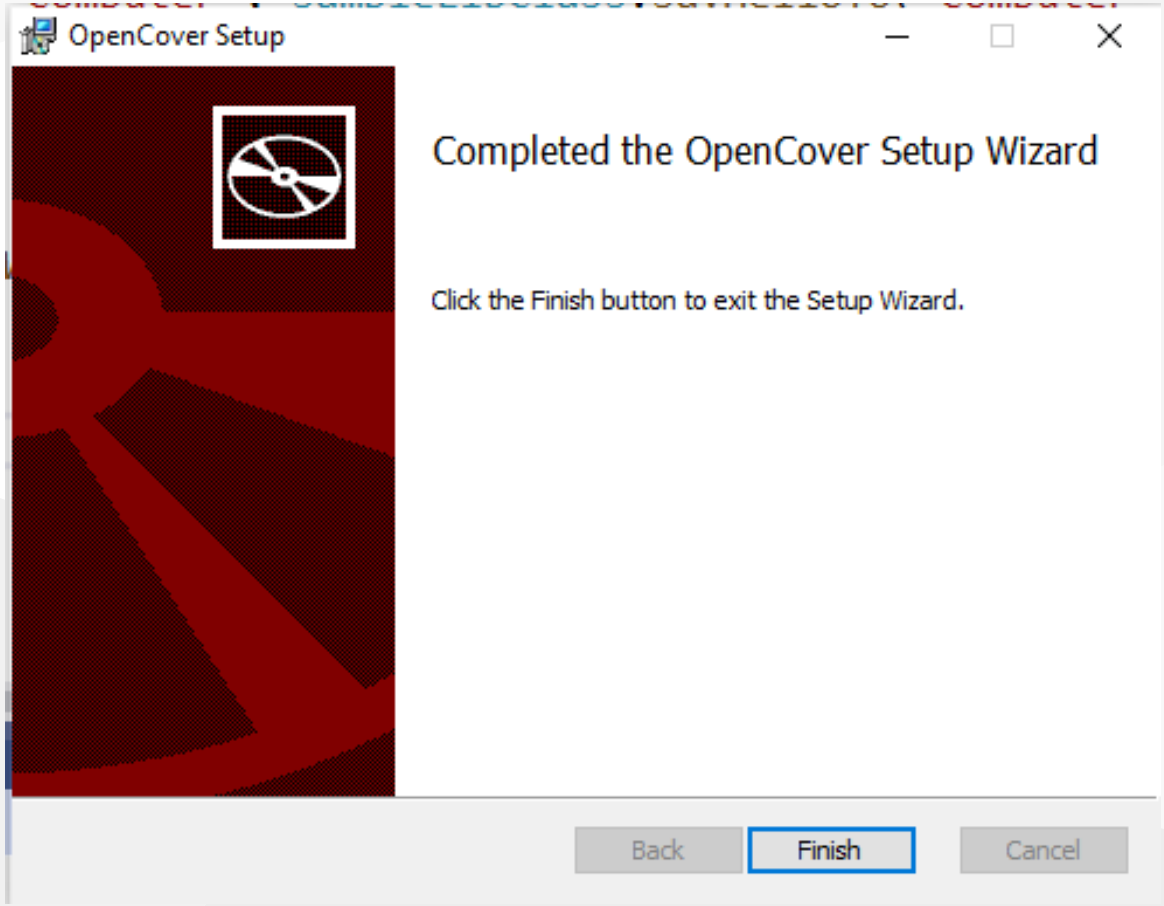
Visual Studio Community Edition (C# Unit Test+OpenCover + Nunit Runner + Report)-11



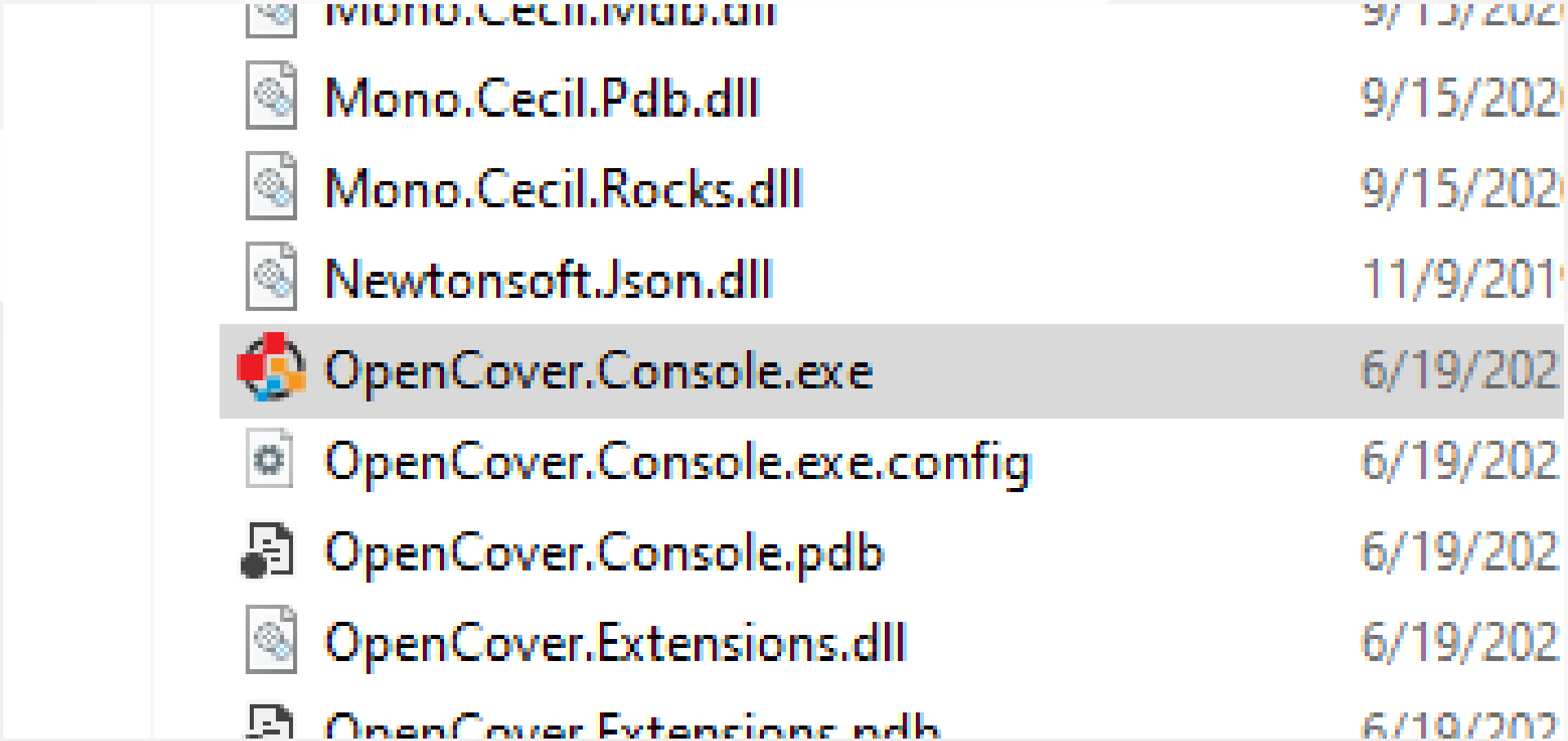
Visual Studio Community Edition (C# Unit Test+OpenCover + Nunit Runner + Report)-12



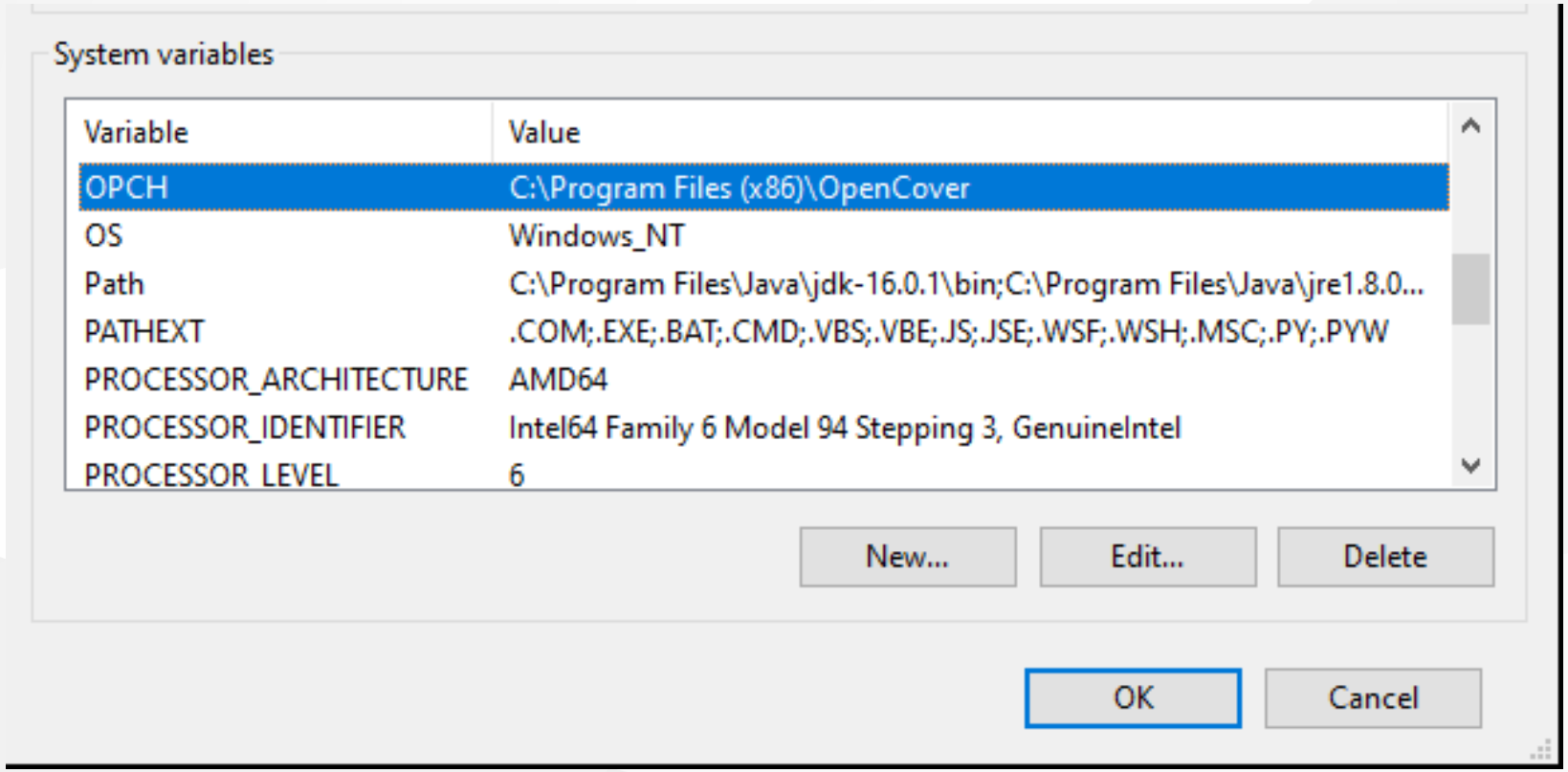
Visual Studio Community Edition (C# Unit Test+OpenCover + Nunit Runner + Report)-13

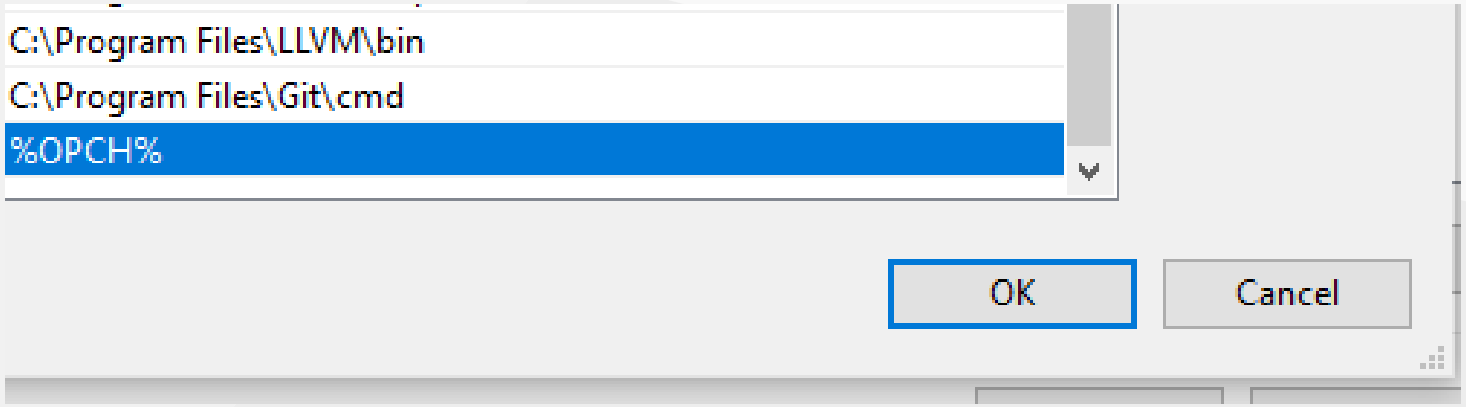


Visual Studio Community Edition (C# Unit Test+OpenCover + Nunit Runner + Report)-14



Visual Studio Community Edition (C# Unit Test+OpenCover + Nunit Runner + Report)-15





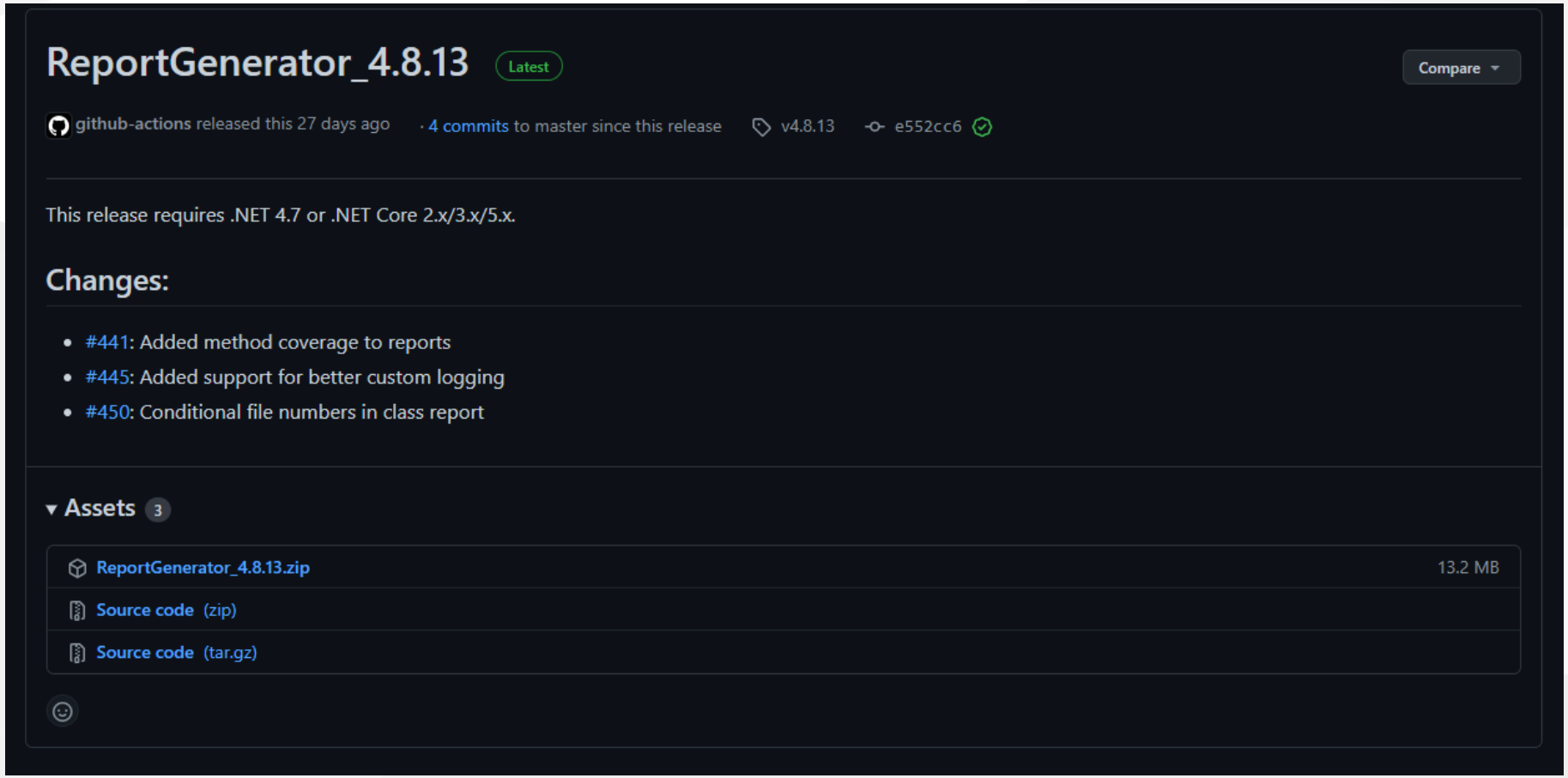
Visual Studio Community Edition (C# Unit Test+OpenCover + Nunit Runner + Report)-16

```
Microsoft Windows [Version 10.0.19043.1288]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ugur.coruh>OpenCover.Console
Launching OpenCover 4.7.1221.0

Incorrect Arguments: The target argument is required

Usage:
    ["]-target:<target application>["]
    [["]-targetdir:<target directory>["]]
    [["]-searchdirs:<additional PDB directory>[;<additional PDB
    [["]-targetargs:<arguments for the target process>["]]
```

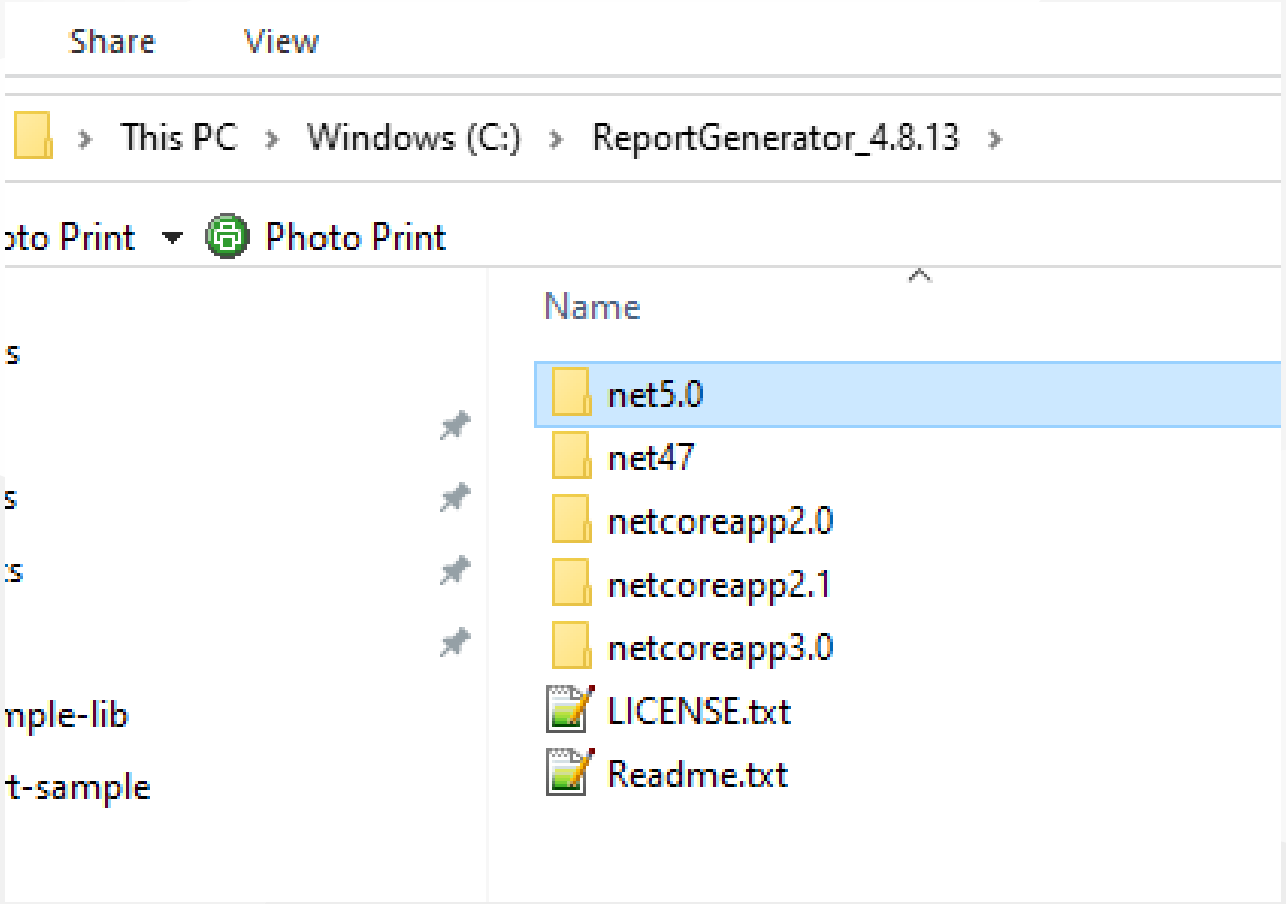
Visual Studio Community Edition (C# Unit Test+OpenCover + Nunit Runner + Report)-17

ReportGenerator

- Release ReportGenerator_4.8.13 · danielpalme/ReportGenerator · GitHub



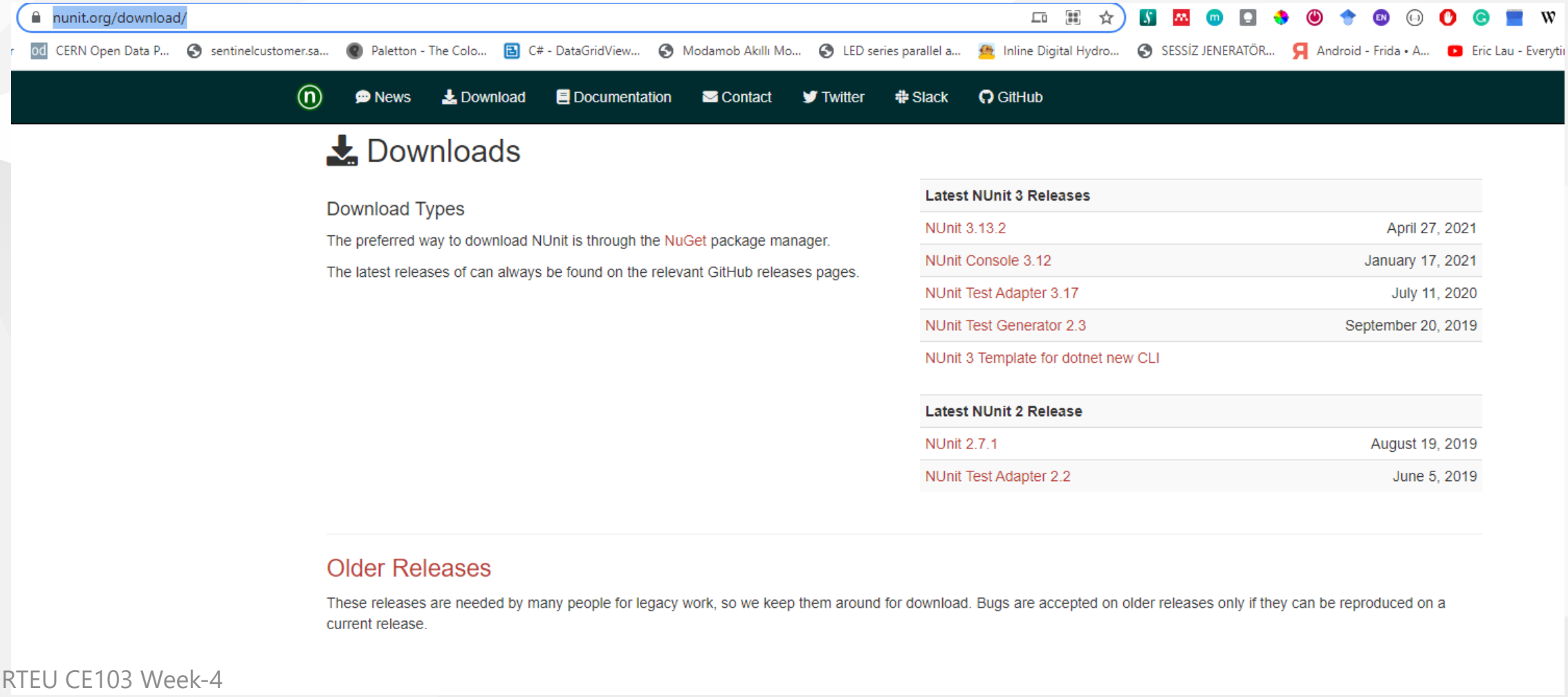
Visual Studio Community Edition (C# Unit Test+OpenCover + Nunit Runner + Report)-18



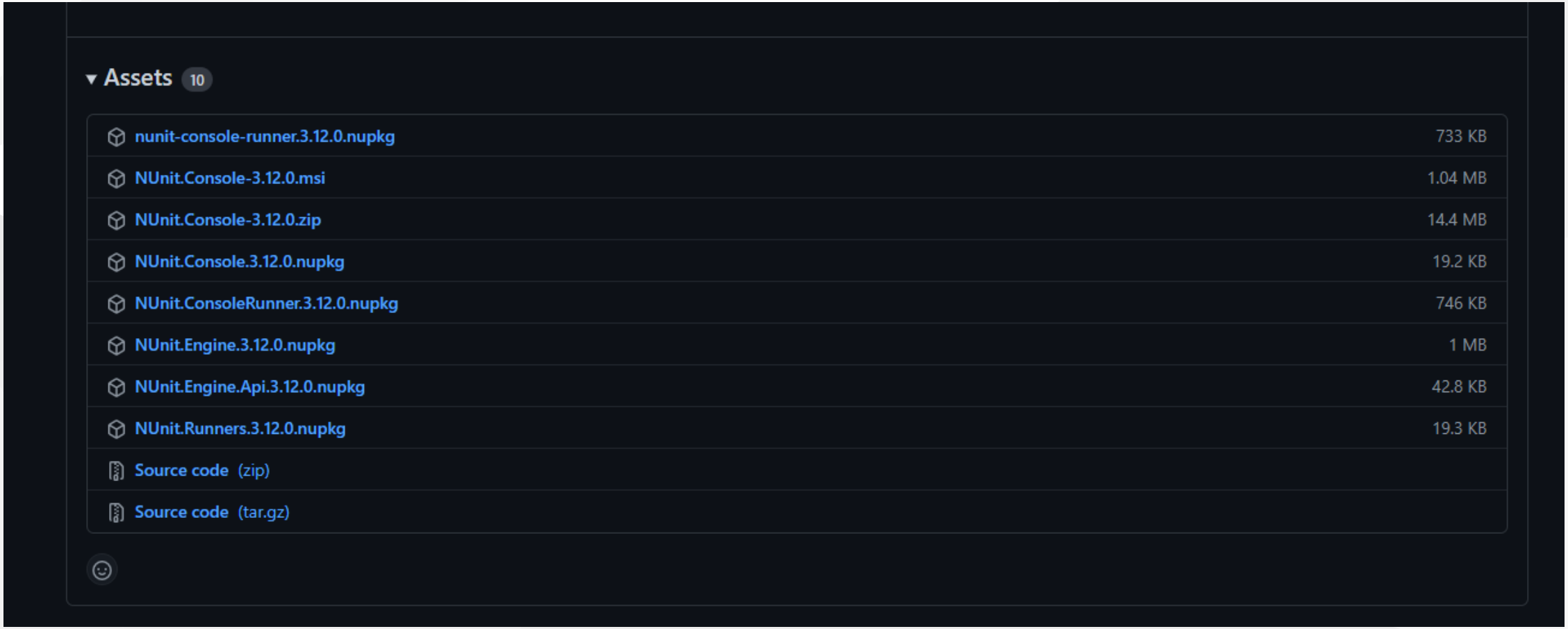
Visual Studio Community Edition (C# Unit Test+OpenCover + Nunit Runner + Report)-19

NUnit Console

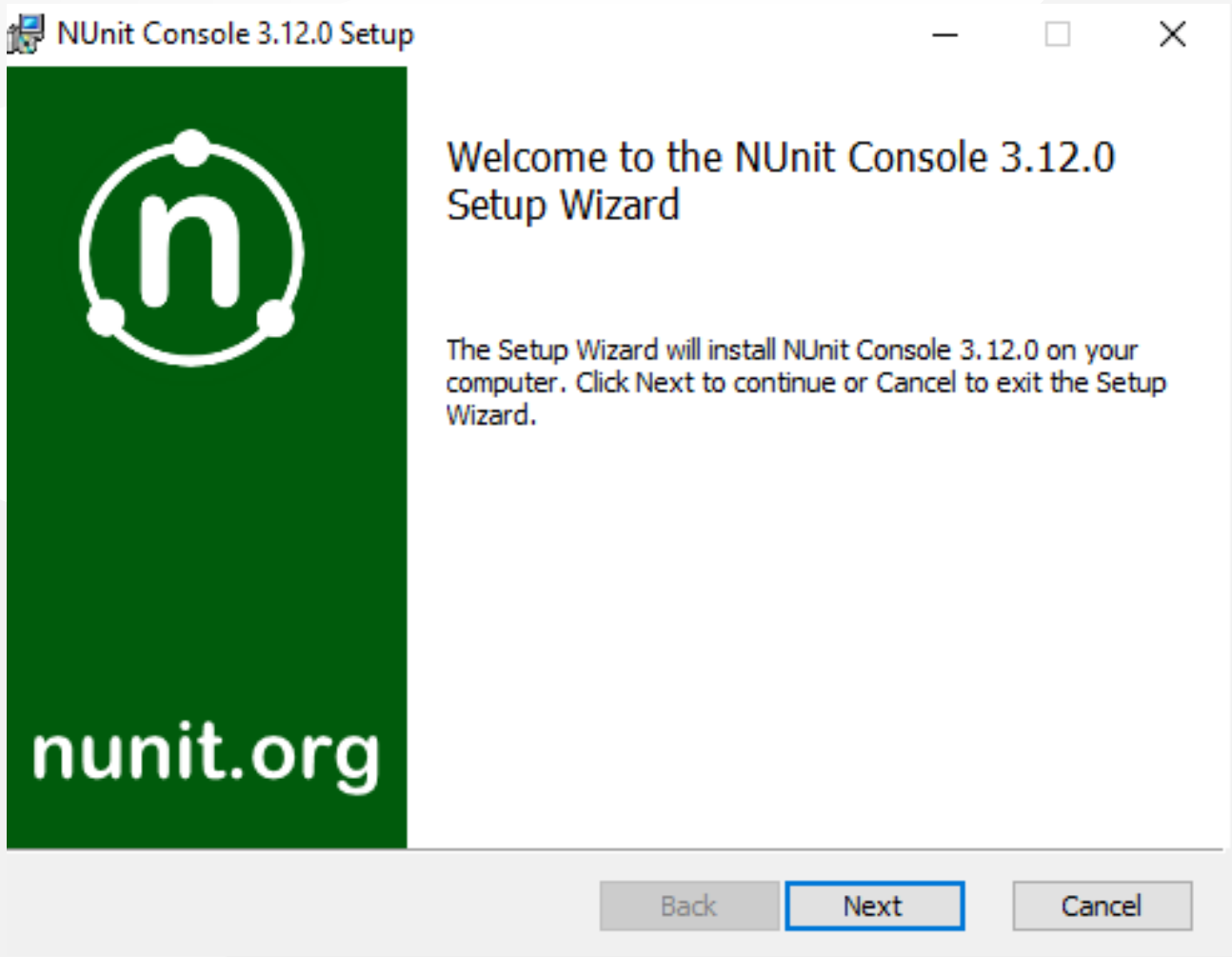
- Downloads



Visual Studio Community Edition (C# Unit Test+OpenCover + Nunit Runner + Report)-20

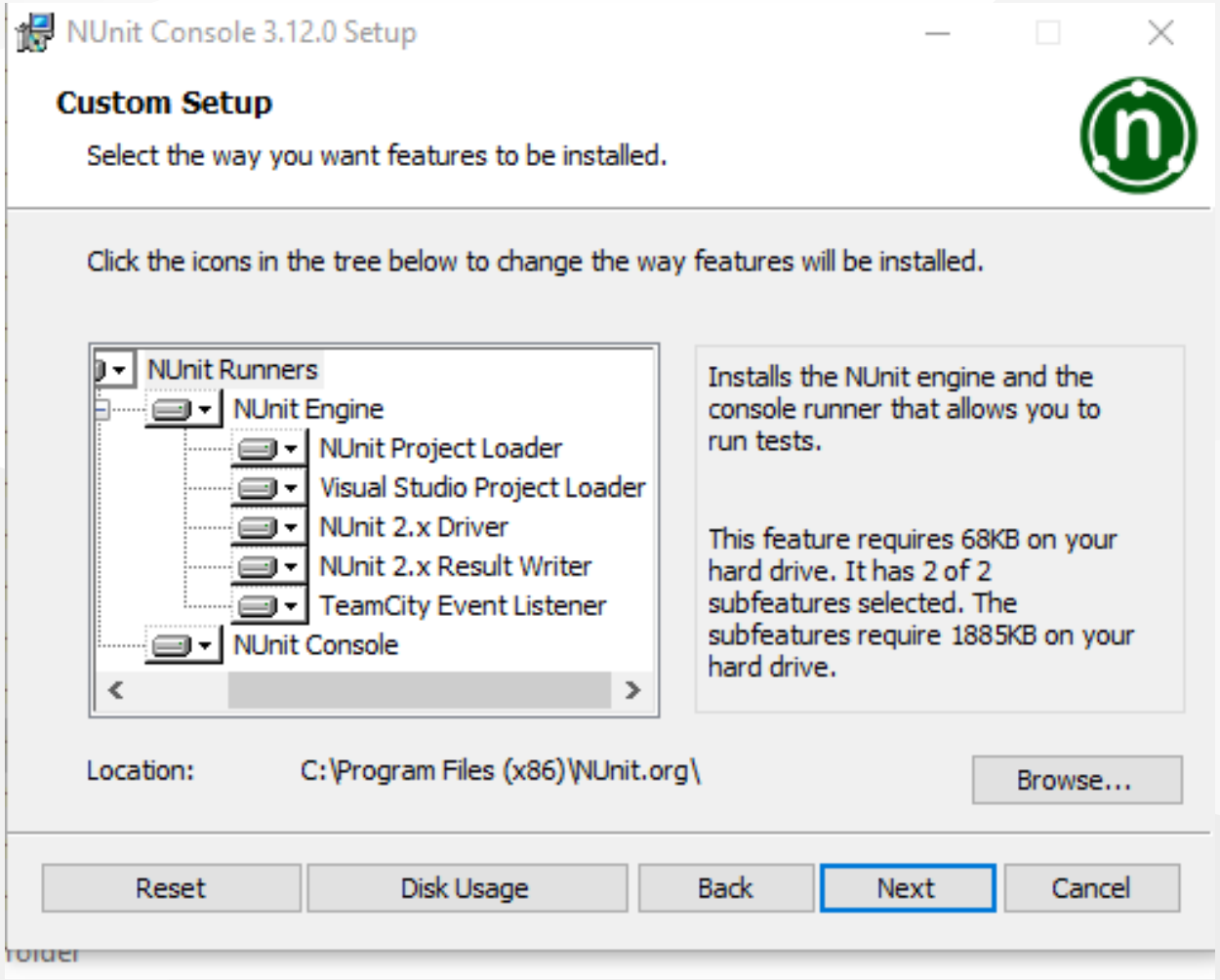


Visual Studio Community Edition (C# Unit Test+OpenCover + Nunit Runner + Report)-21



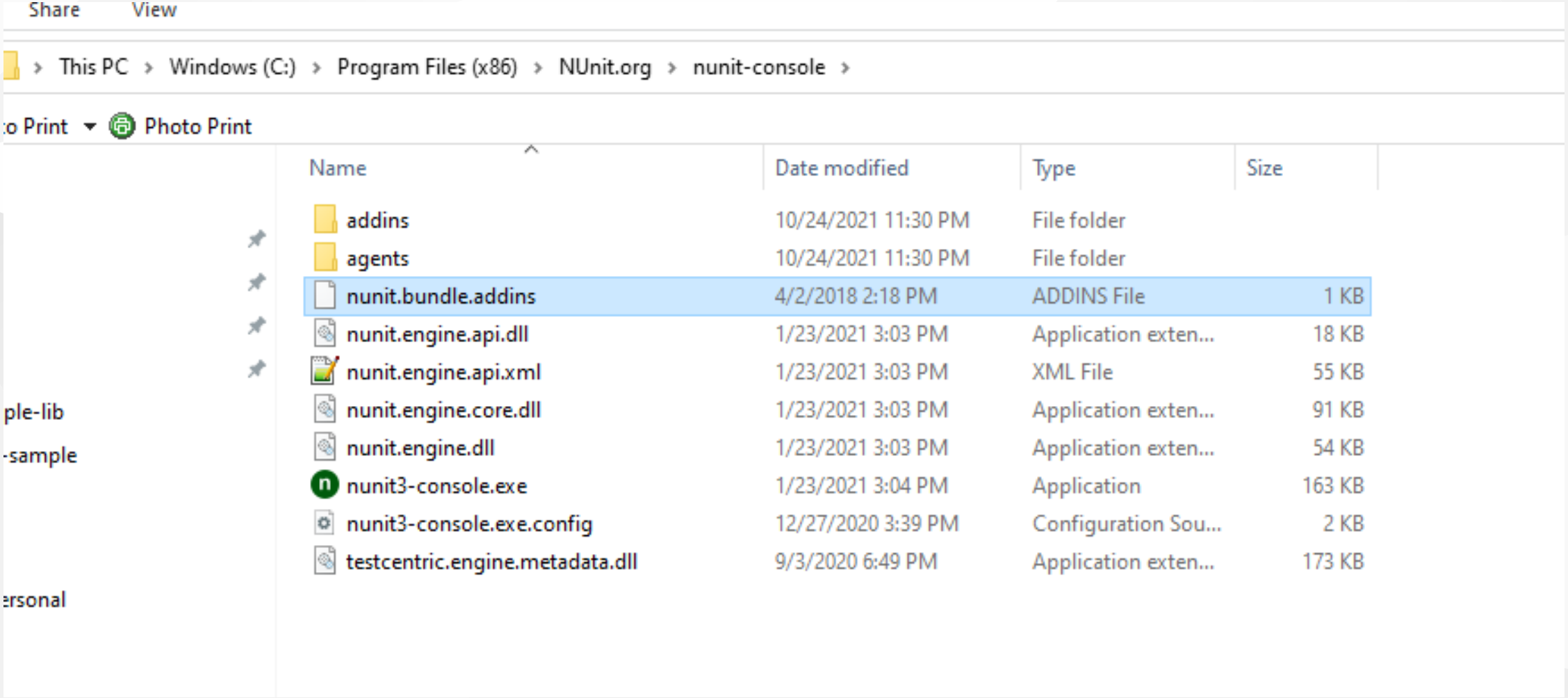
Visual Studio Community Edition (C# Unit Test+OpenCover + Nunit Runner + Report)-22

- Download setup



Visual Studio Community Edition (C# Unit Test+OpenCover + Nunit Runner + Report)-23

- Install setup



Visual Studio Community Edition (C# Unit Test+OpenCover + Nunit Runner + Report)-24

NUnit + MSTest Batch Report Generation (Not Tested)

- OpenCover and ReportGenerator Unit Test Coverage in Visual Studio 2013 and 2015 – CodeHelper.Net
- OpenCover and ReportGenerator Unit Test Coverage in Visual Studio 2013 and 2015 - CodeProject

Java Unit Tests

Eclipse IDE (JUnit4 , JUnit5)

Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test

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

JUnit 5

JUnit 5 User Guide

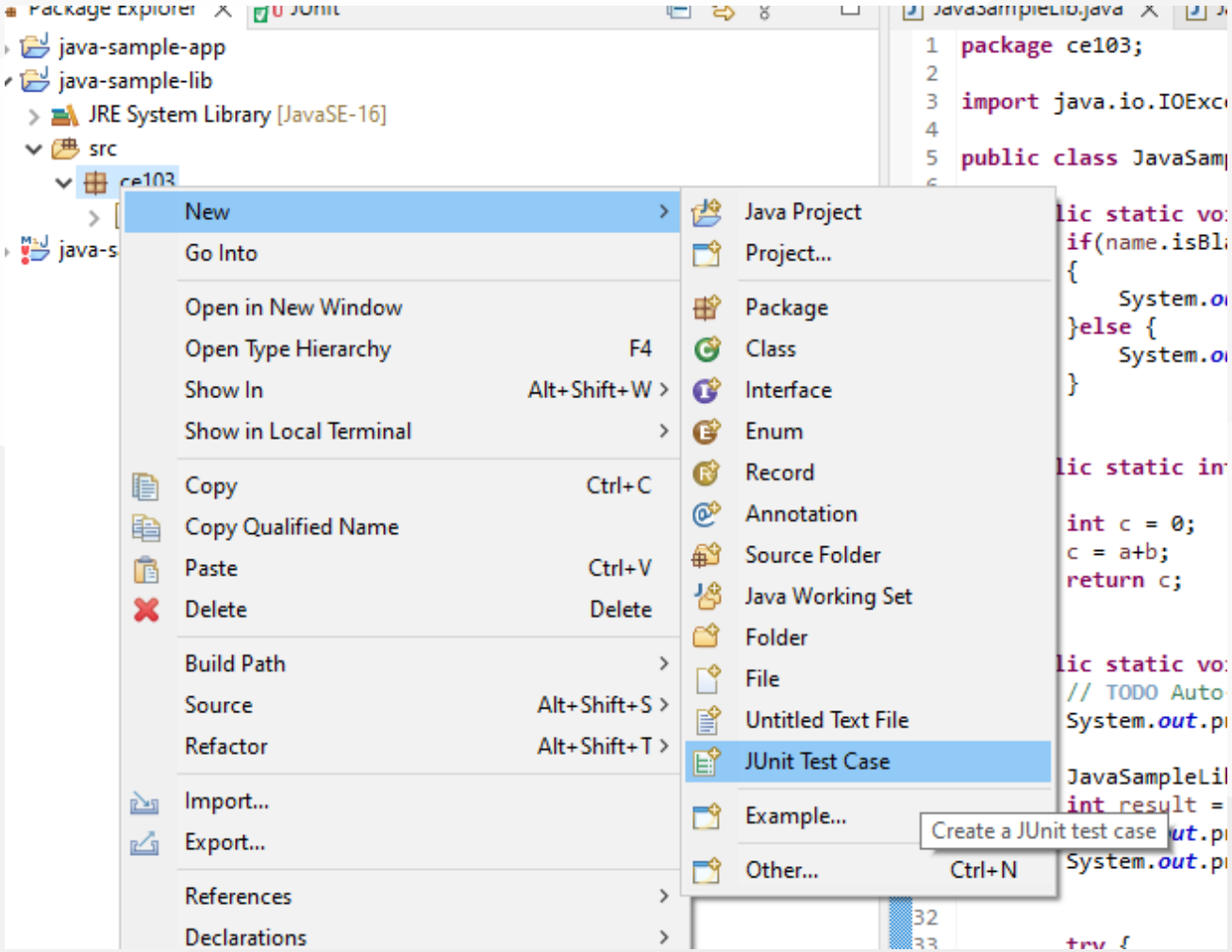
https://www.eclemma.org/

JUnit Hello World Example - Examples Java Code Geeks - 2021

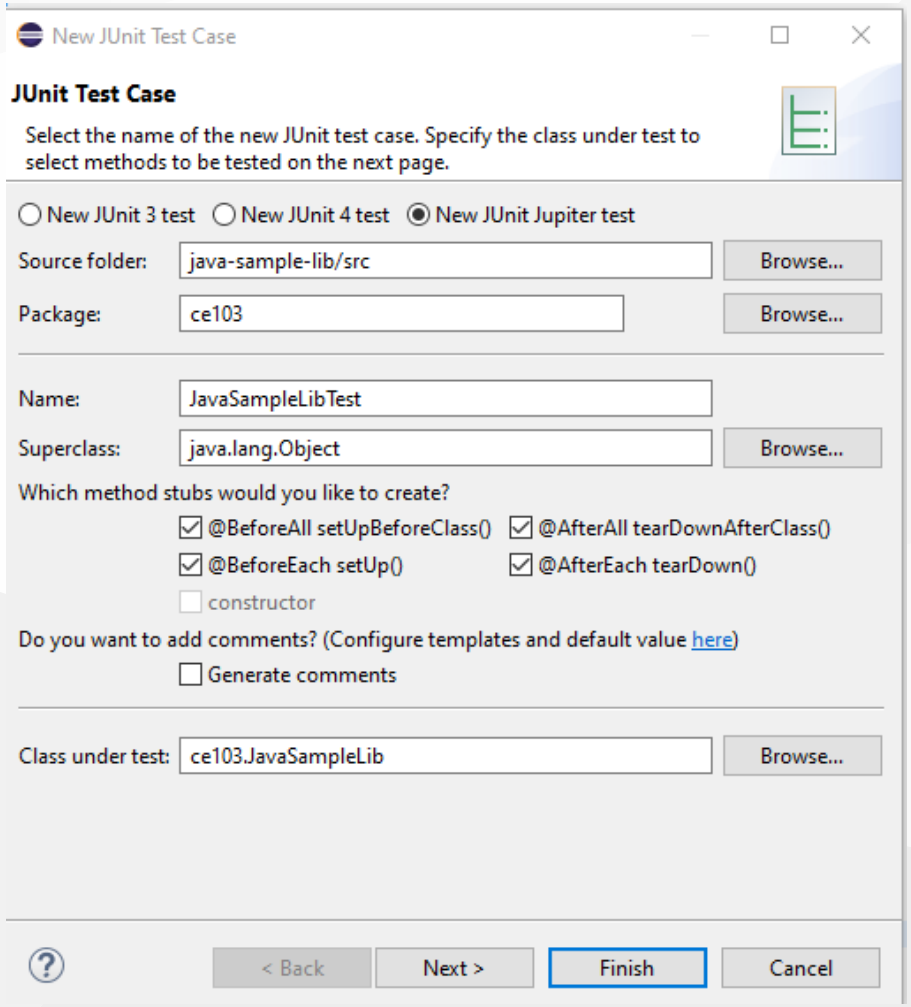
https://yasinmemic.medium.com/java-ile-unit-test-yazmak-birim-test-ca15cf0d024b

Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test

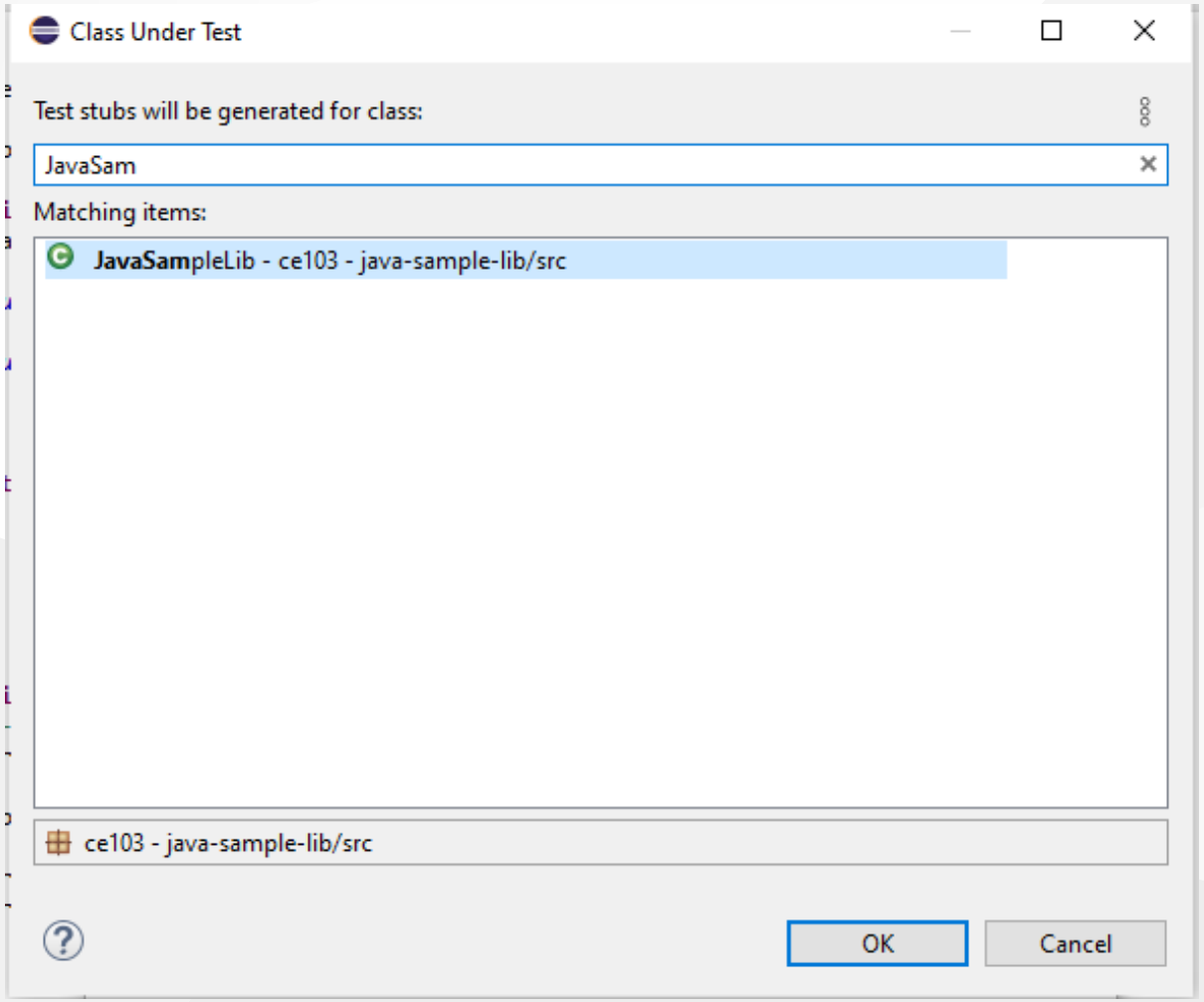
In normal java application we can right click the project java-sample-lib and add Junit case



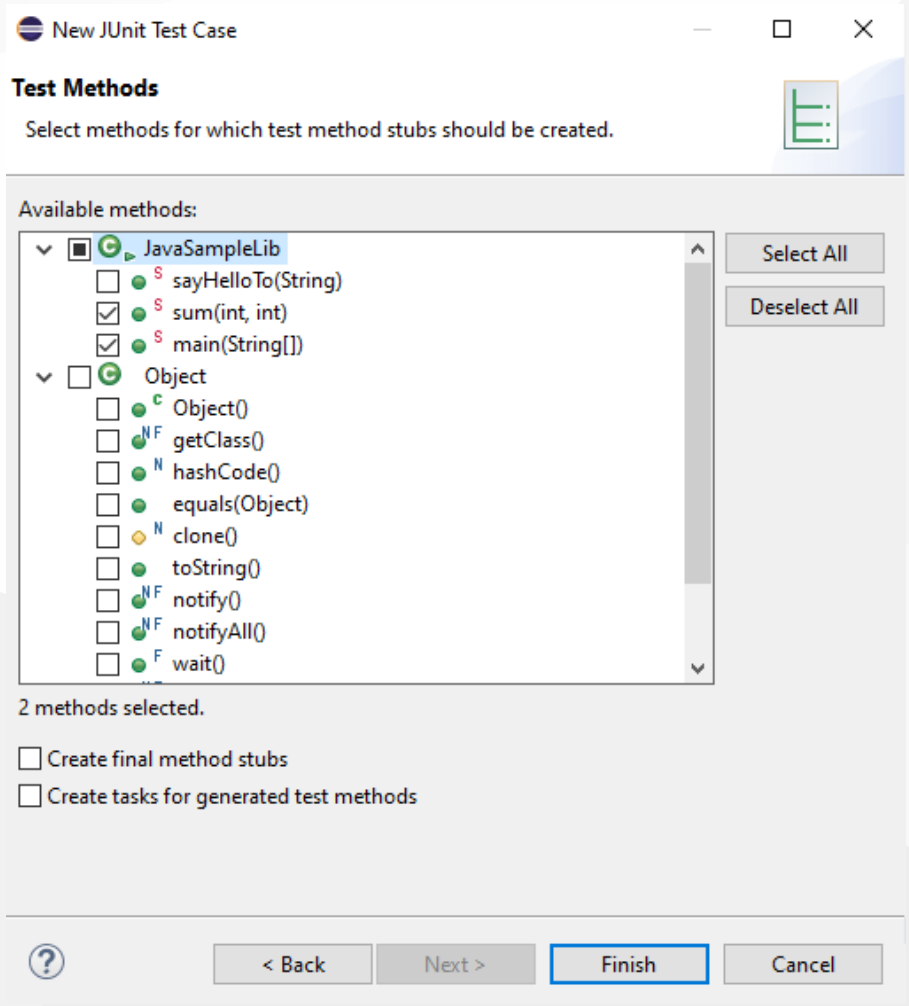
Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test



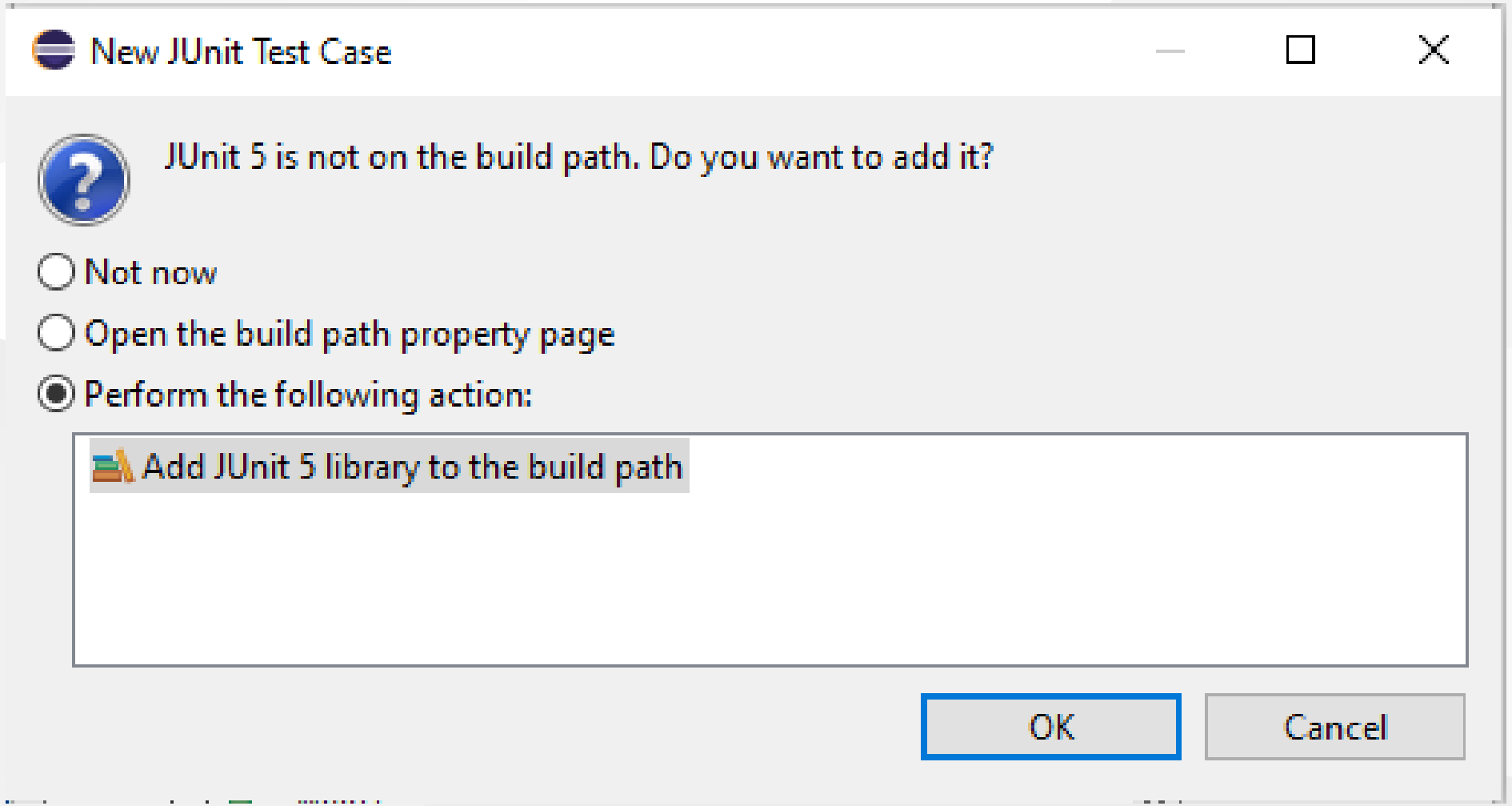
Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test



Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test

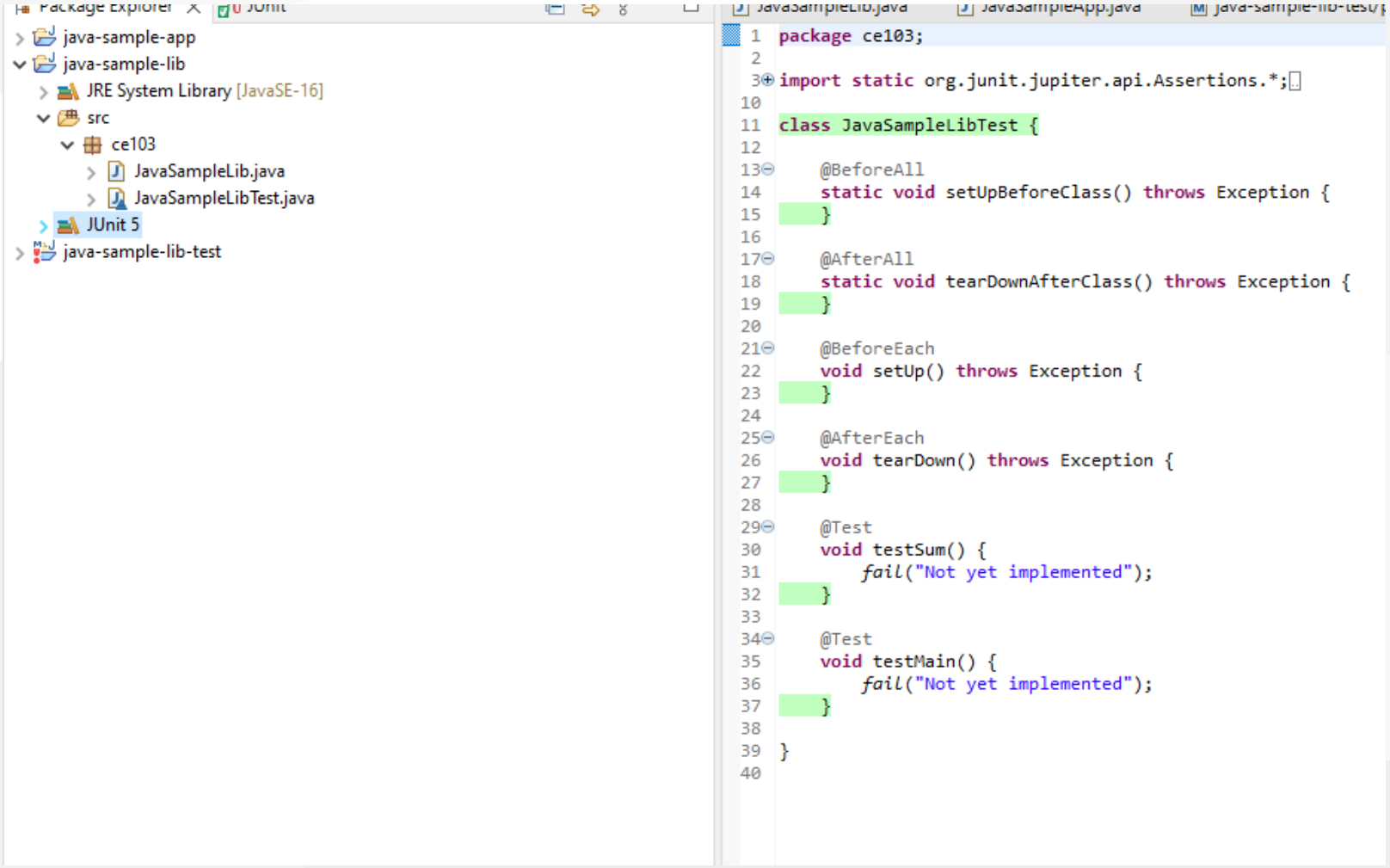


Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test



Eclipse IDE (JUnit4, JUnit5) + Java Unit Test

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 follows to check outputs

JavaSampleLib.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();
//        }
//
```

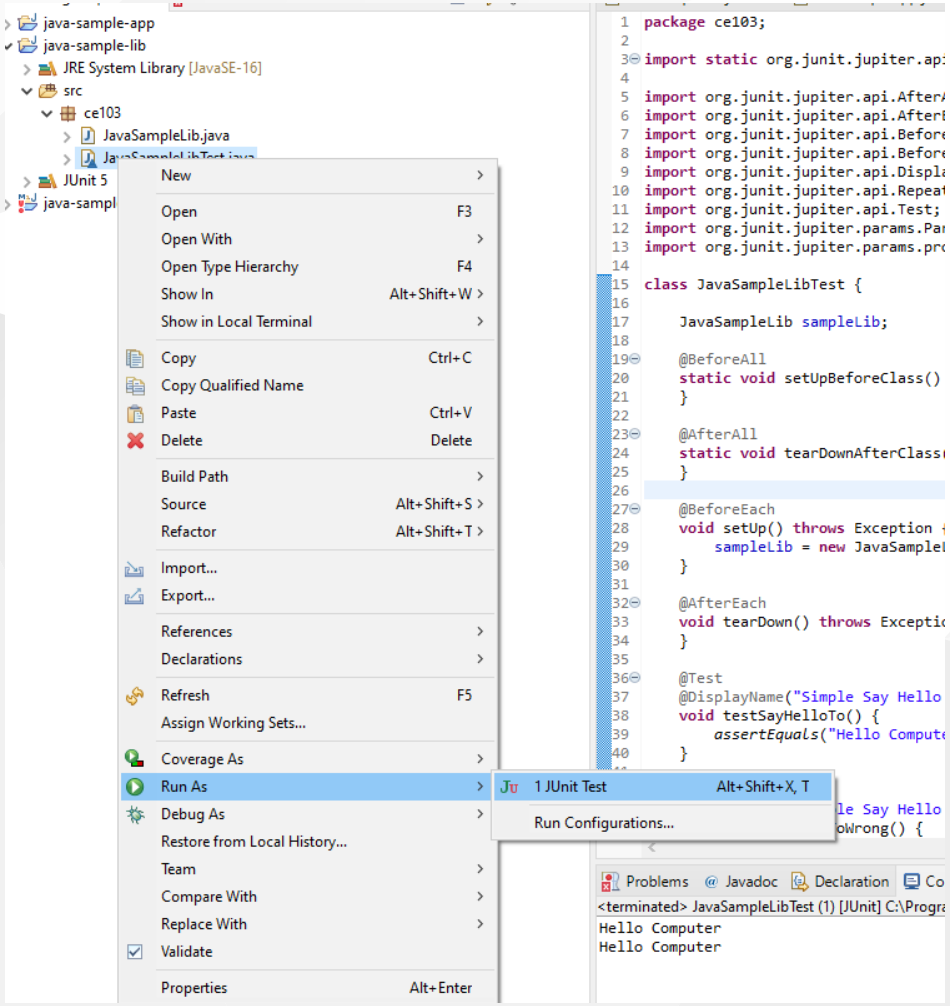
Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test

and JavaSampleLibTest.java

```java
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.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 work");
    }

    @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));
    }

}
```

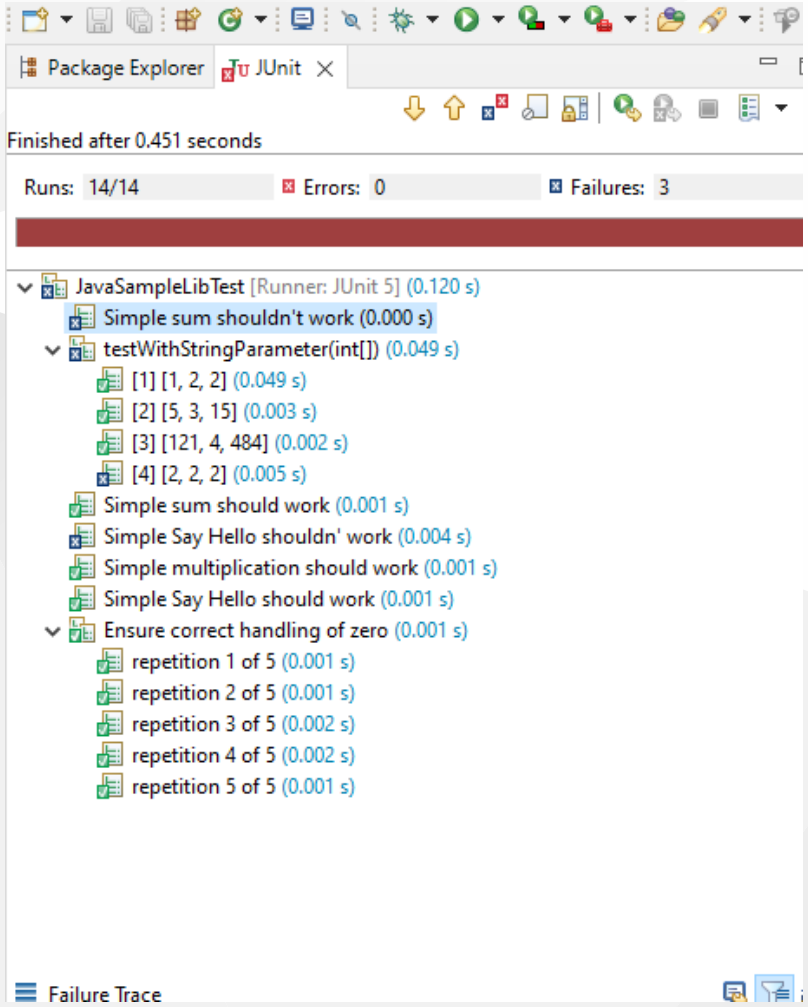
Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test

if we run tests



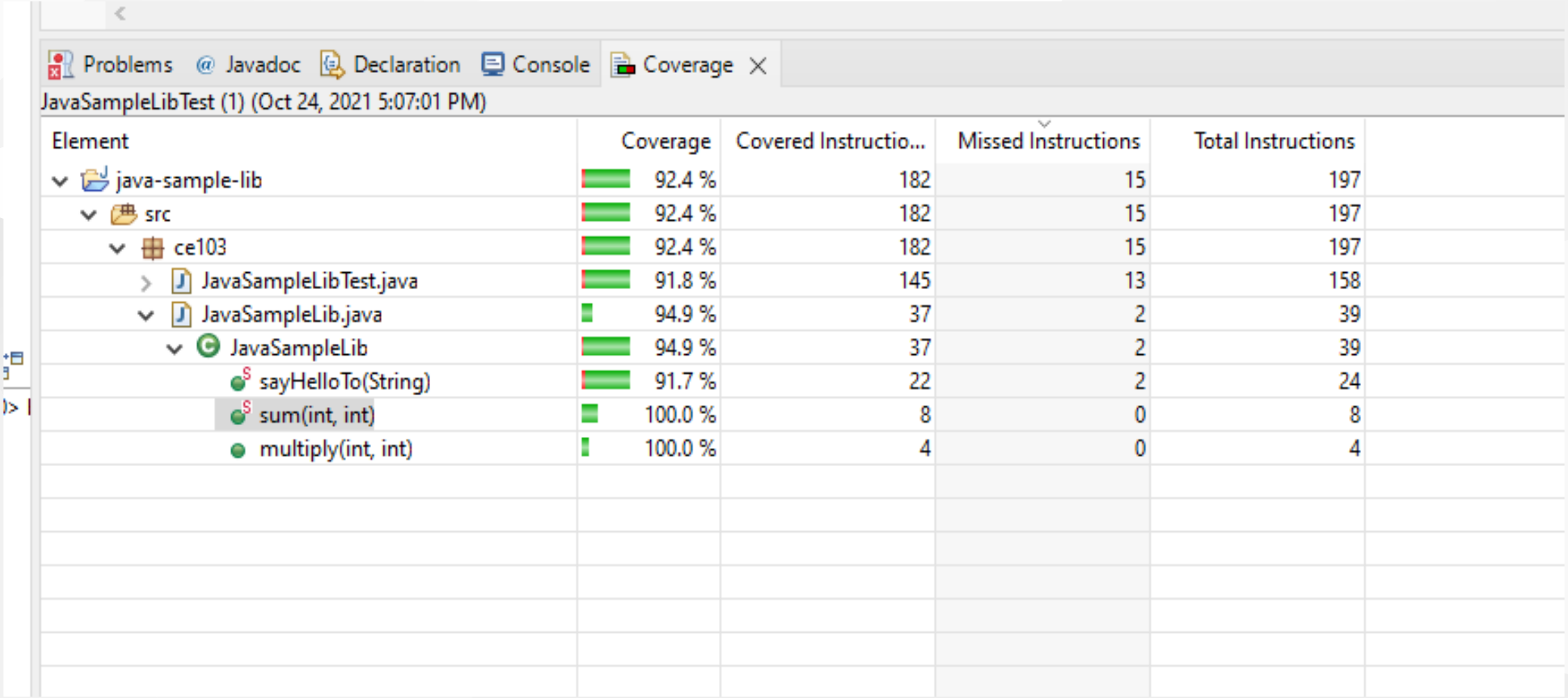
Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test

we will see all results there



Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test

also we can see the code coverage of tests



Element	Coverage	Covered Instructio...	Missed Instructions	Total Instructions
java-sample-lib	92.4 %	182	15	197
src	92.4 %	182	15	197
ce103	92.4 %	182	15	197
JavaSampleLibTest.java	91.8 %	145	13	158
JavaSampleLib.java	94.9 %	37	2	39
JavaSampleLib	94.9 %	37	2	39
sayHelloTo(String)	91.7 %	22	2	24
sum(int, int)	100.0 %	8	0	8
multiply(int, int)	100.0 %	4	0	4

Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test

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

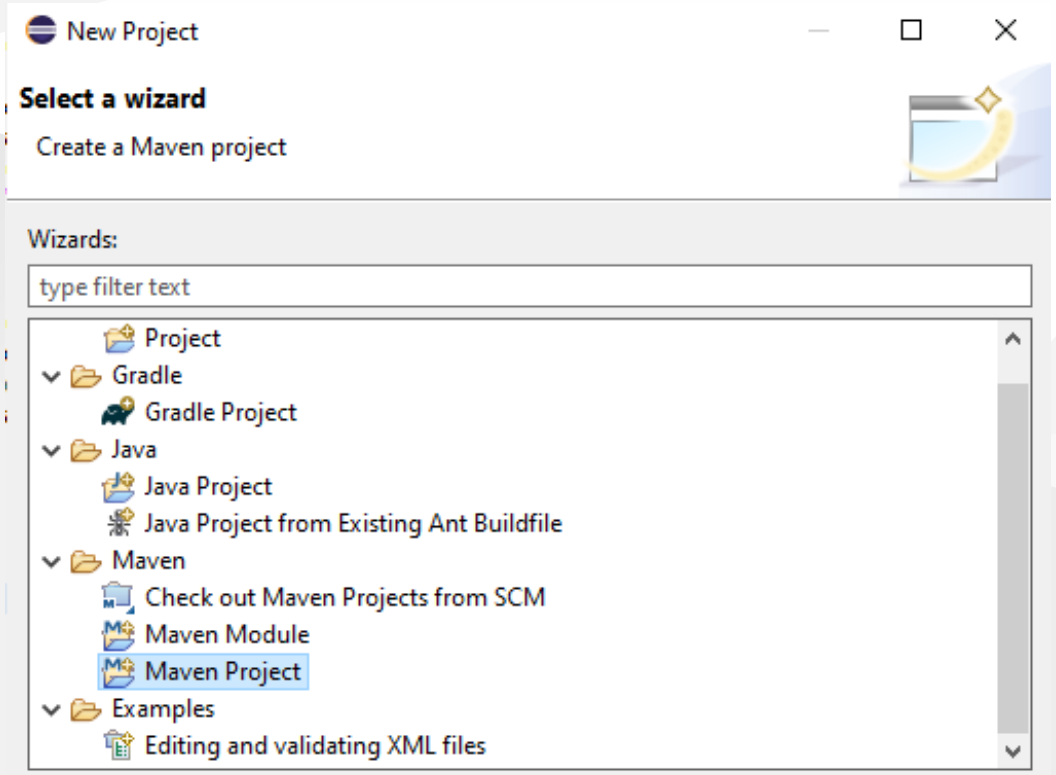
```java
package ce103;

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;
    }

}
```

Maven Java Application + JUnit

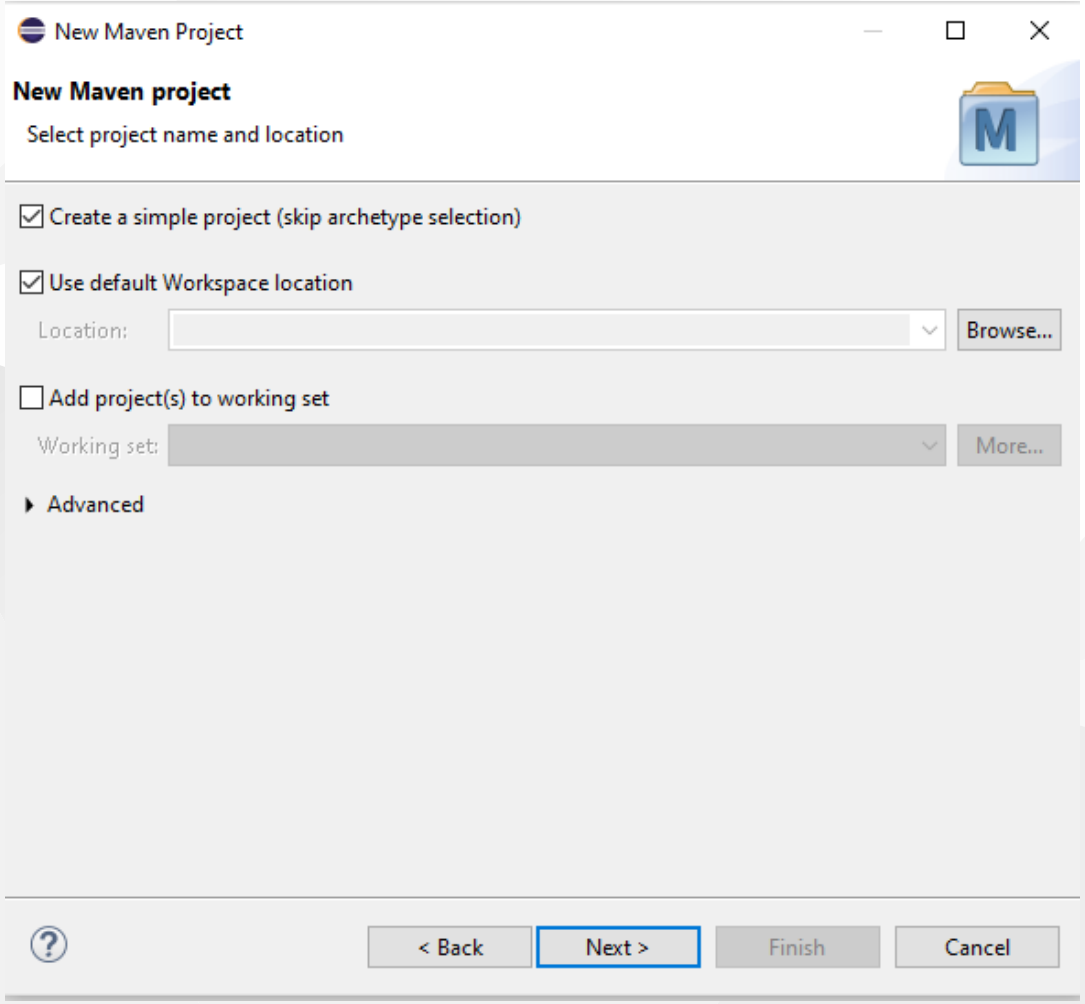
Lets create Maven project with tests

Create a maven project

File -> New -> Maven Project



Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test



Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test

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

Maven – Introduction to the Standard Directory Layout

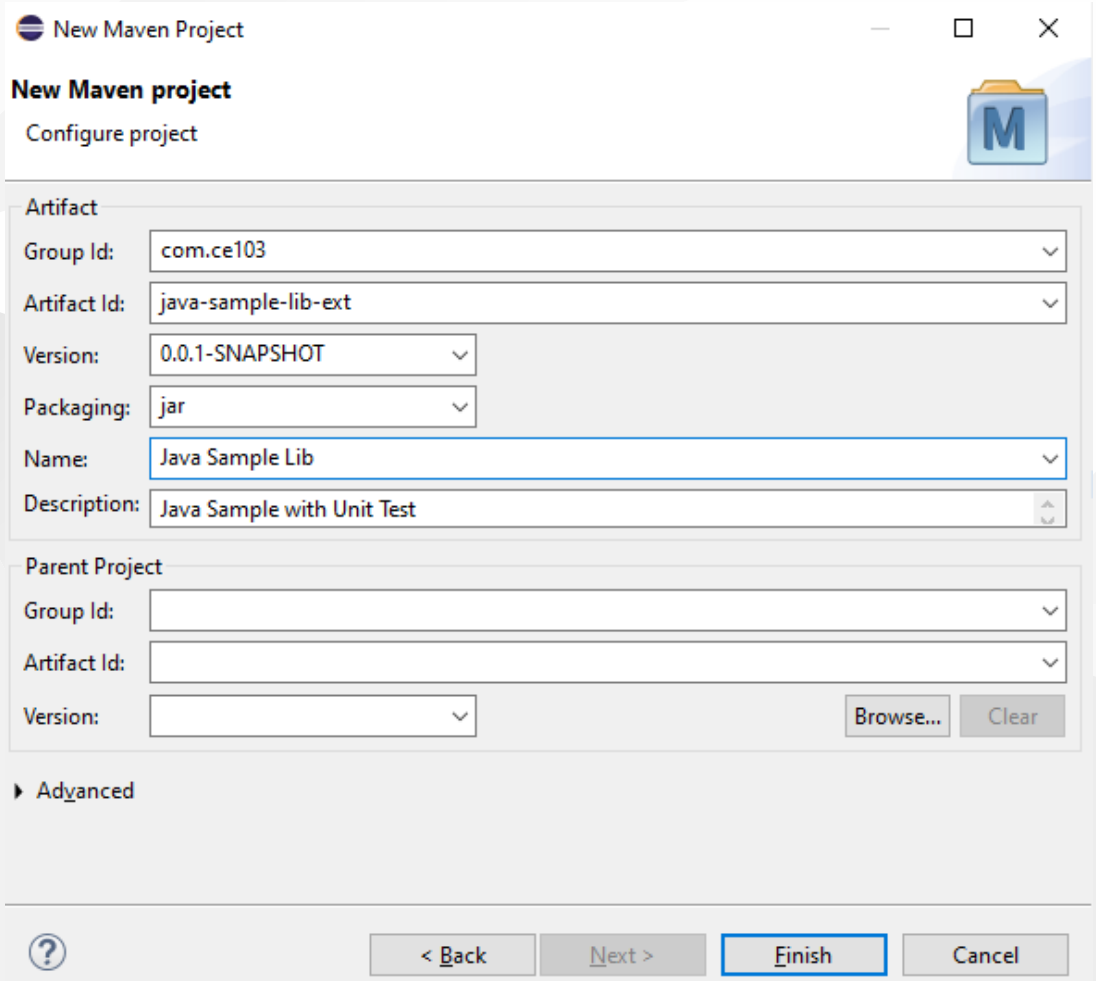
Also for intro you can use this

JUnit Hello World Example - Examples Java Code Geeks - 2021

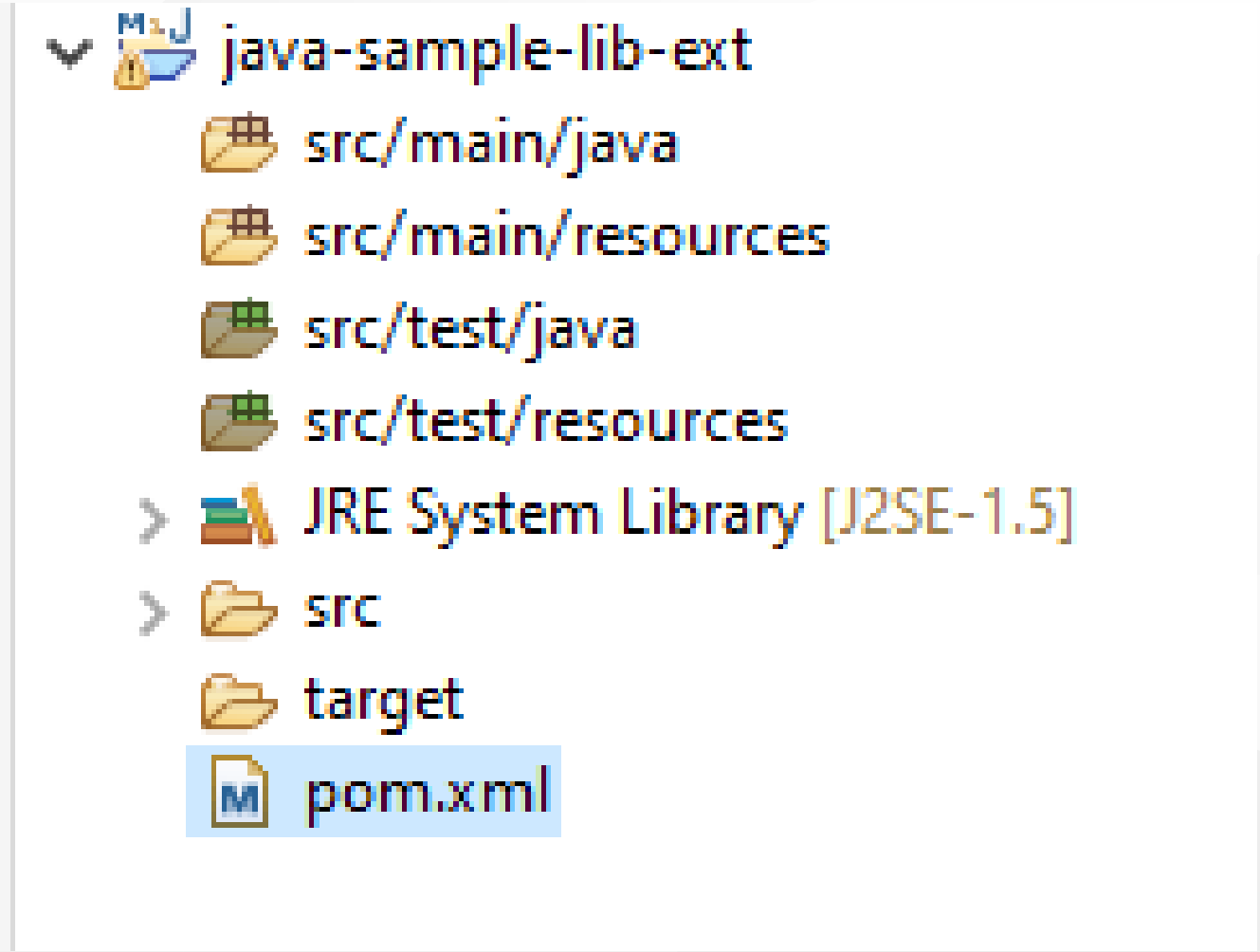
Eclipse
Maven
Java
JUnit 4.12 (pulled by Maven automatically)

Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test

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



Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test



Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test

pom.xml file

```
<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>
</project>
```

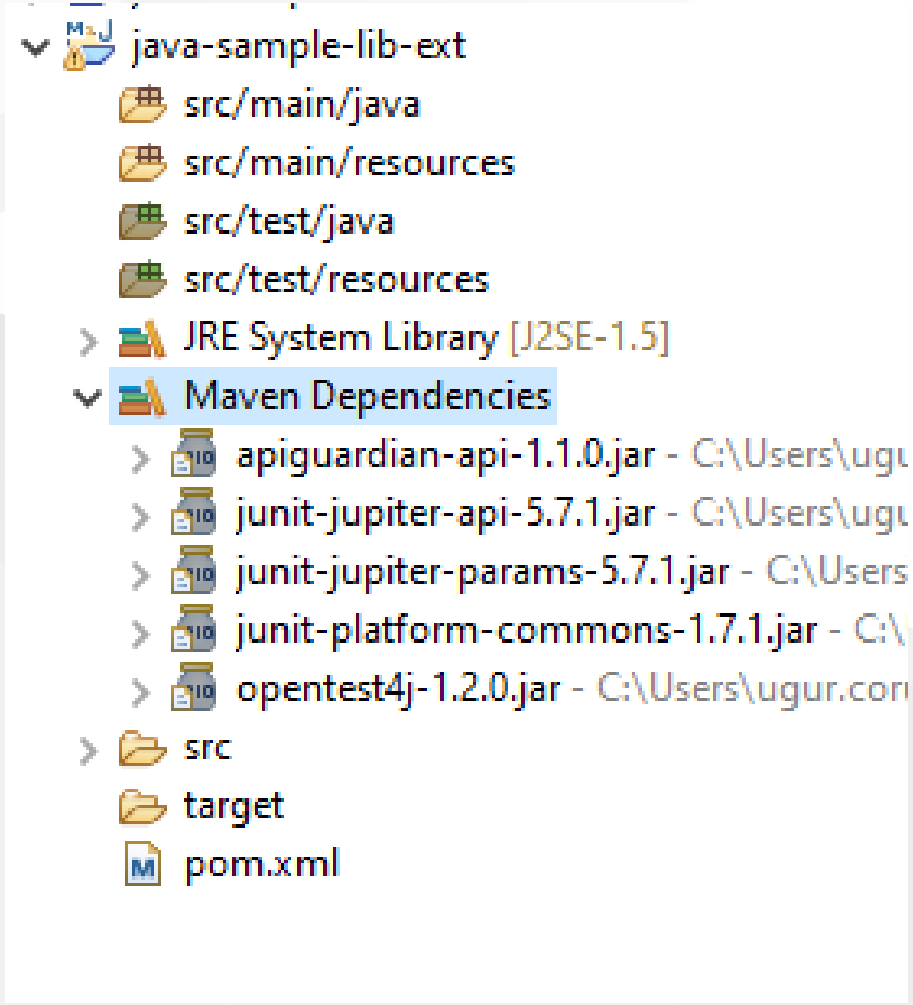
Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test

we will add JUnit 5 for our project

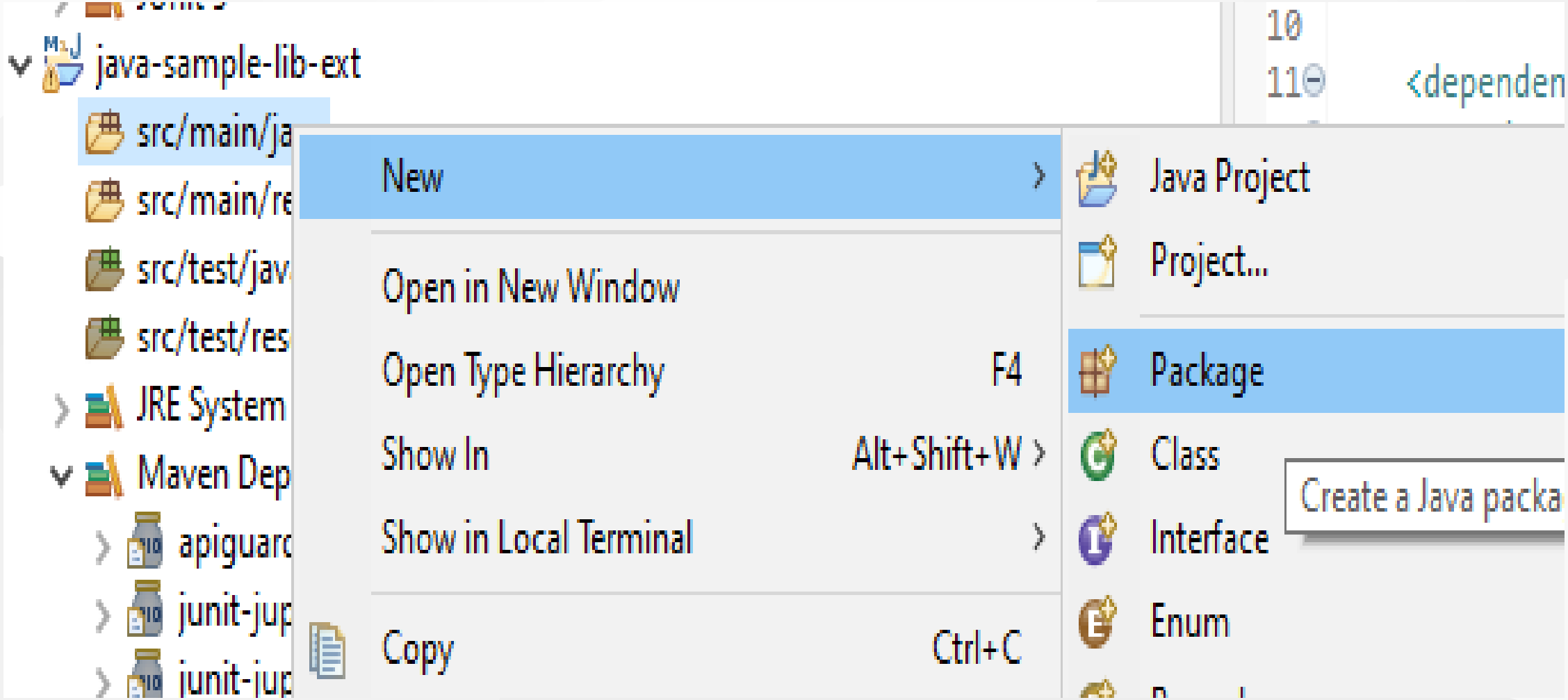
```
<project xmlns="http://maven.apache.org/POM/4.0.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.ce103</groupId>
    <artifactId>java-sample-lib-ext</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>
```

Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test

it will automatically download libraries

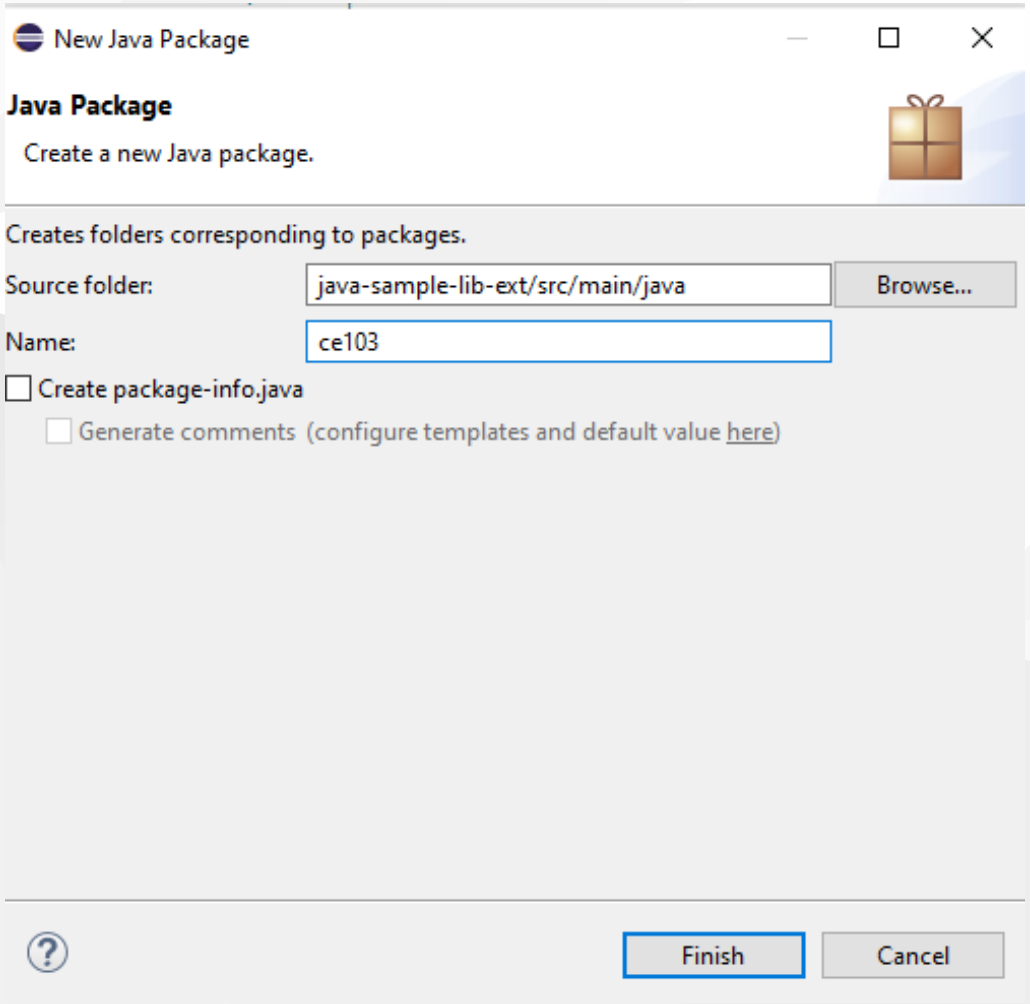


Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test



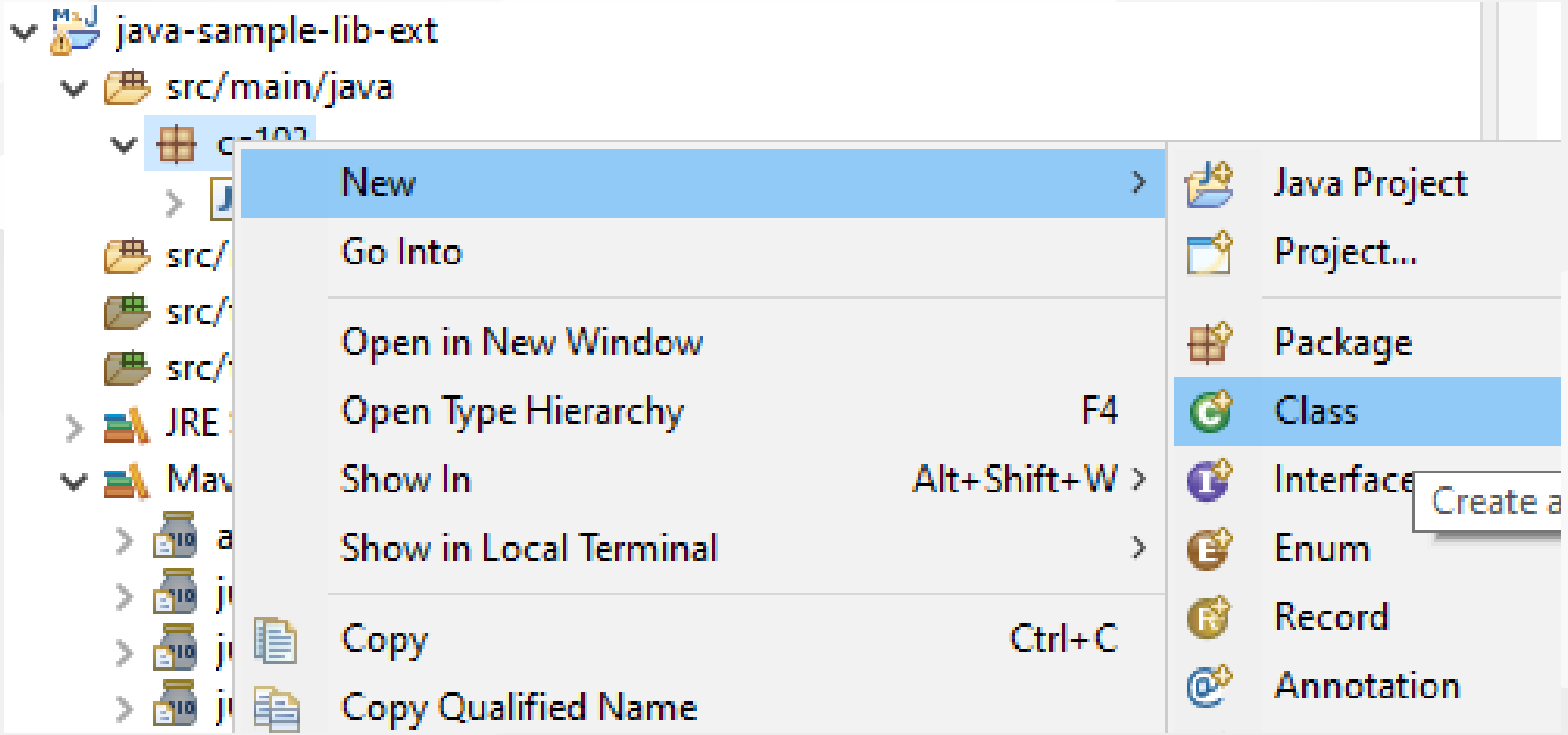
Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test

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

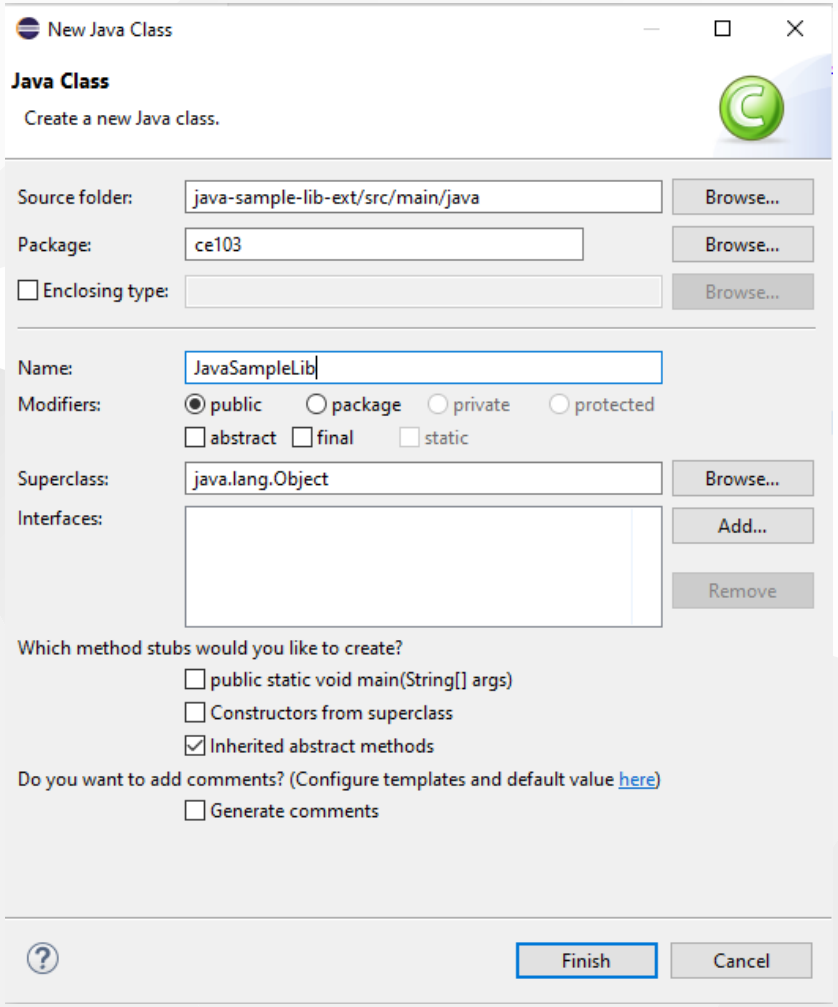


Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test

In this package create library class



Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test



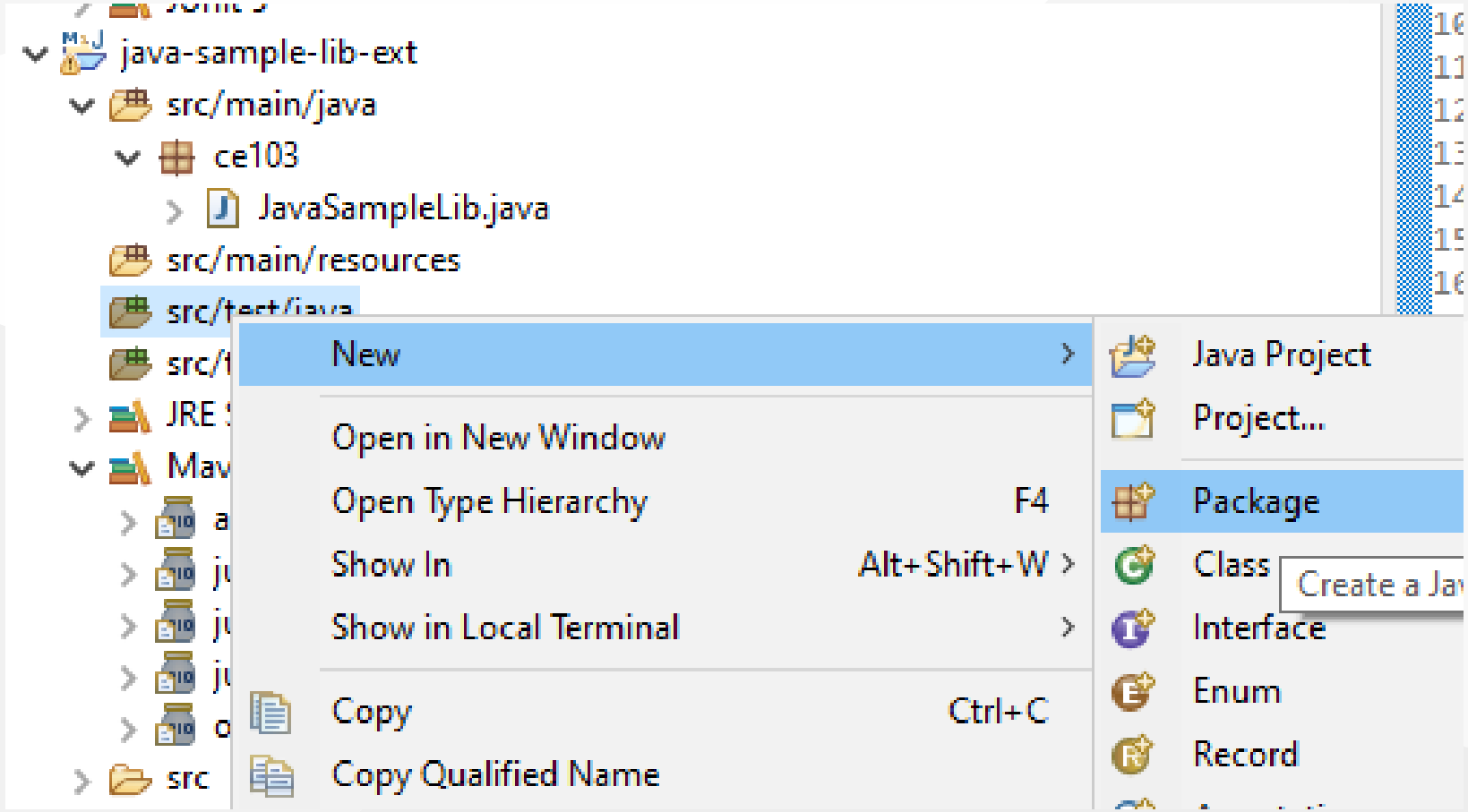
Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test

copy content from other library

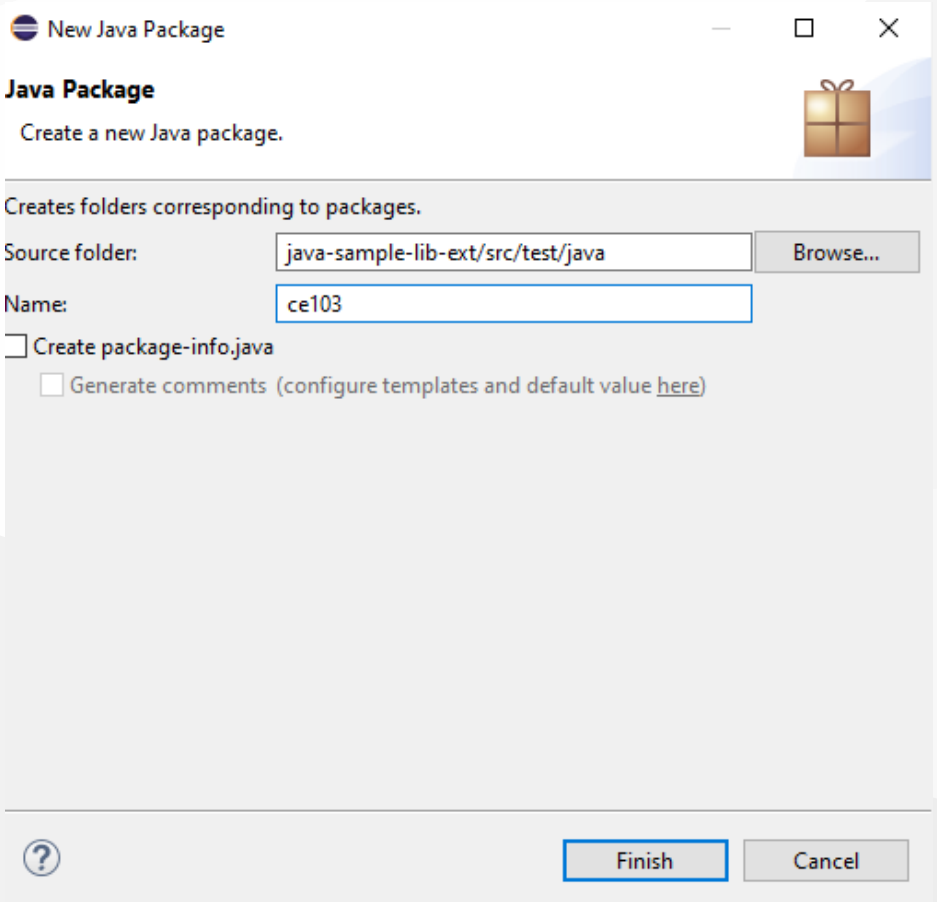
```
package ce103;

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;
    }


}
```

Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test

Now lets create tests inf src/test/java

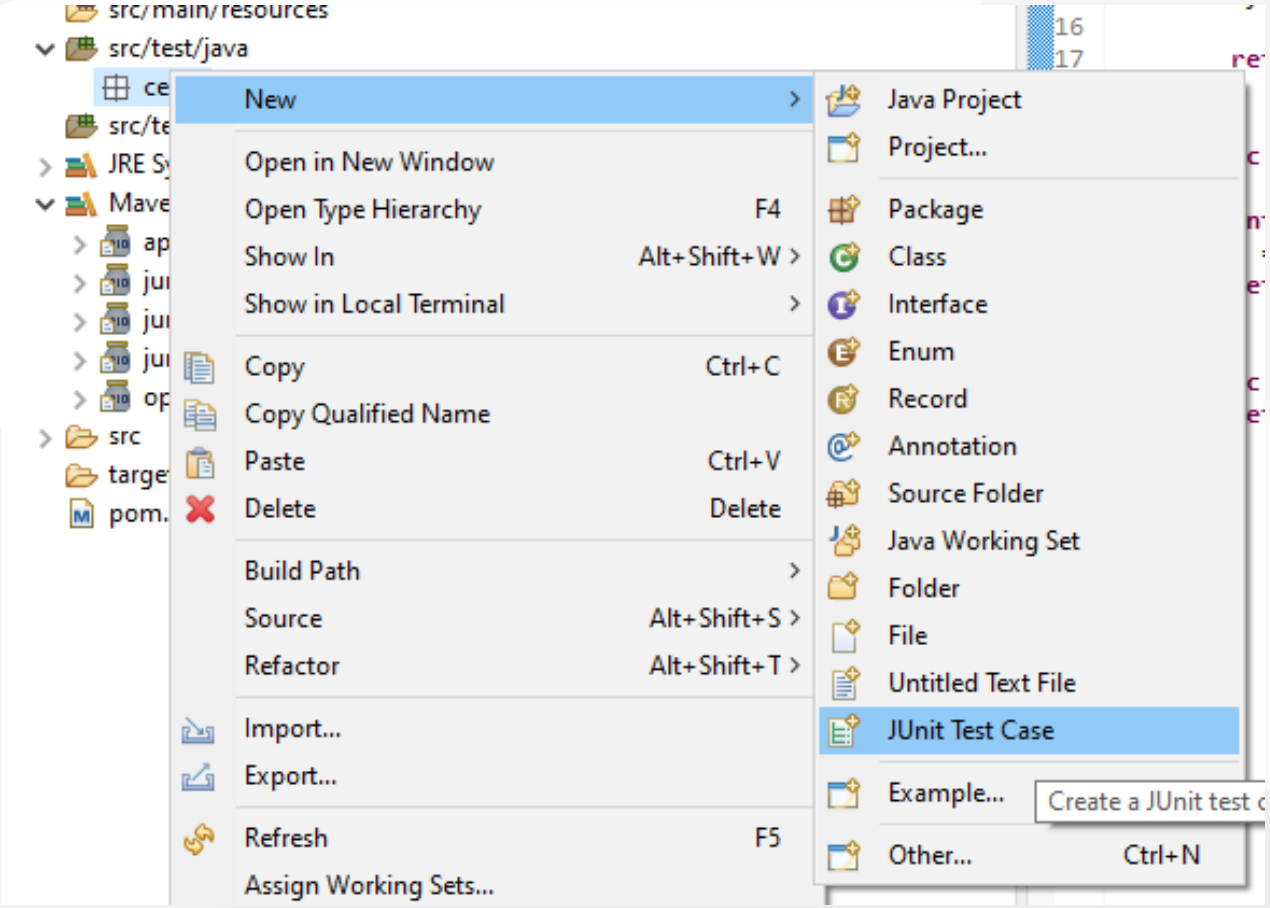


Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test

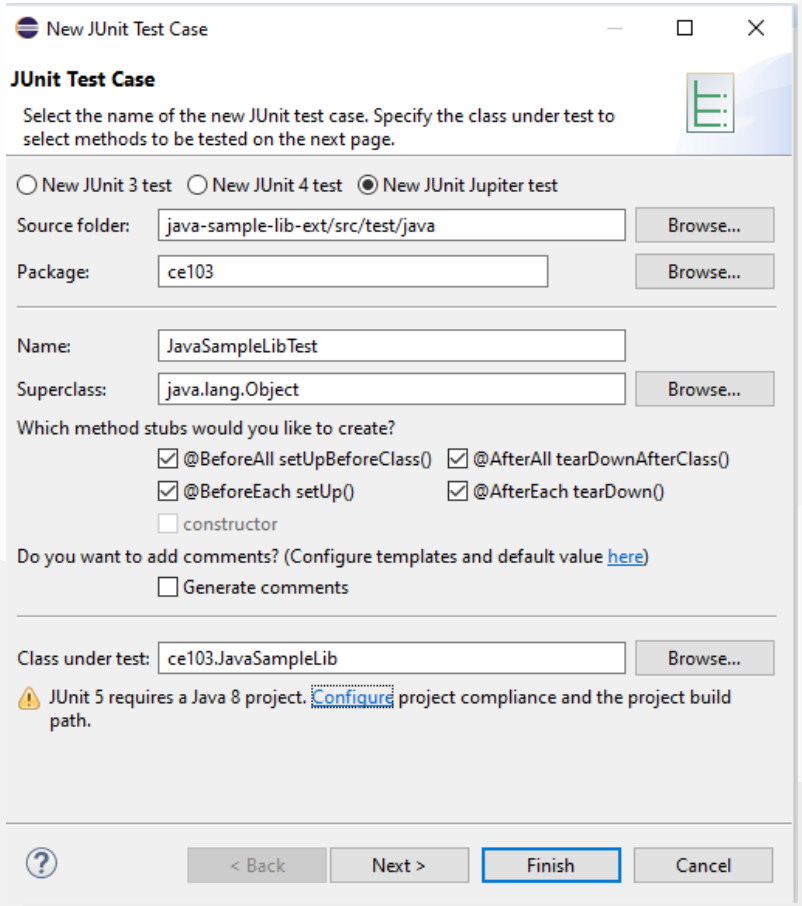


Eclipse IDE (JUnit4 , JUnit5) + Java Unit Test

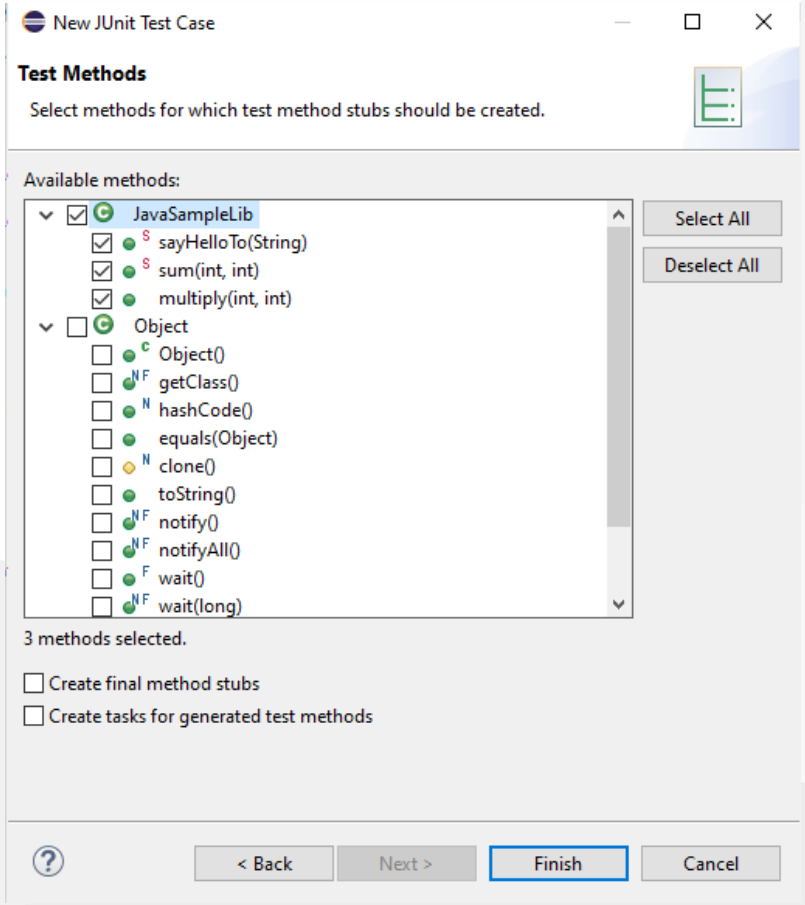
create a JUnit Case



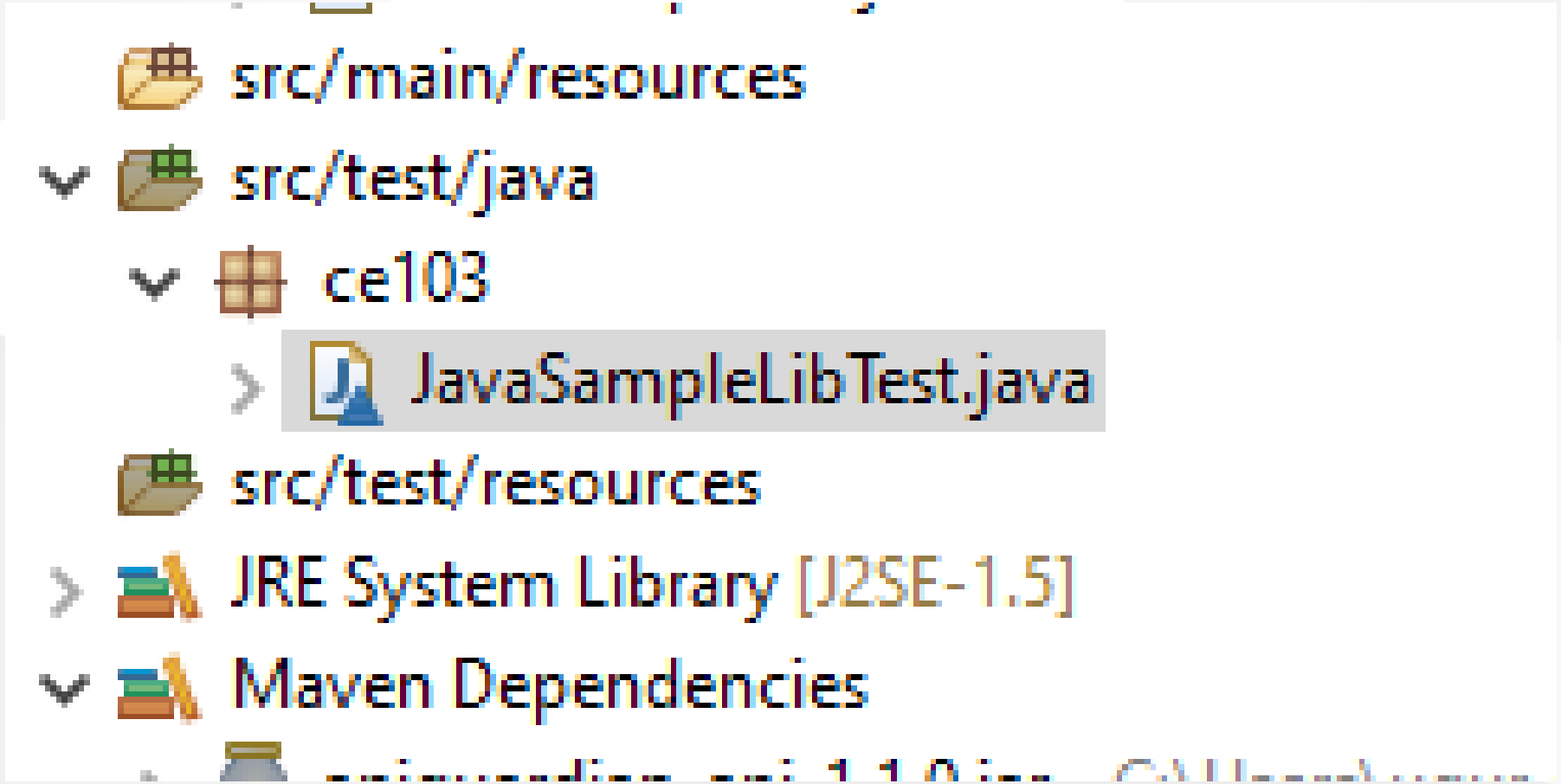
Eclipse IDE (JUnit4, JUnit5) + Java Unit Test



Eclipse IDE (JUnit4, JUnit5) + Java Unit Test



Eclipse IDE (JUnit4, JUnit5) + Java Unit Test



Eclipse IDE (JUnit4, JUnit5) + Java Unit Test

you will simple template

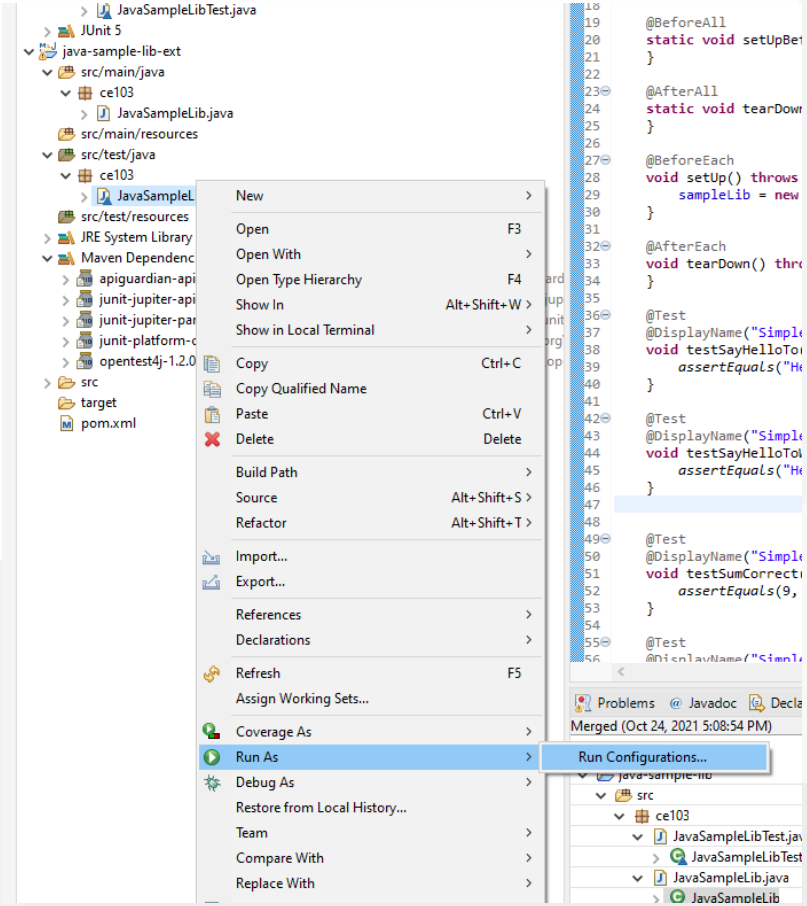
```
package ce103;

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");
    }

}
```

Eclipse IDE (JUnit4, JUnit5) + Java Unit Test

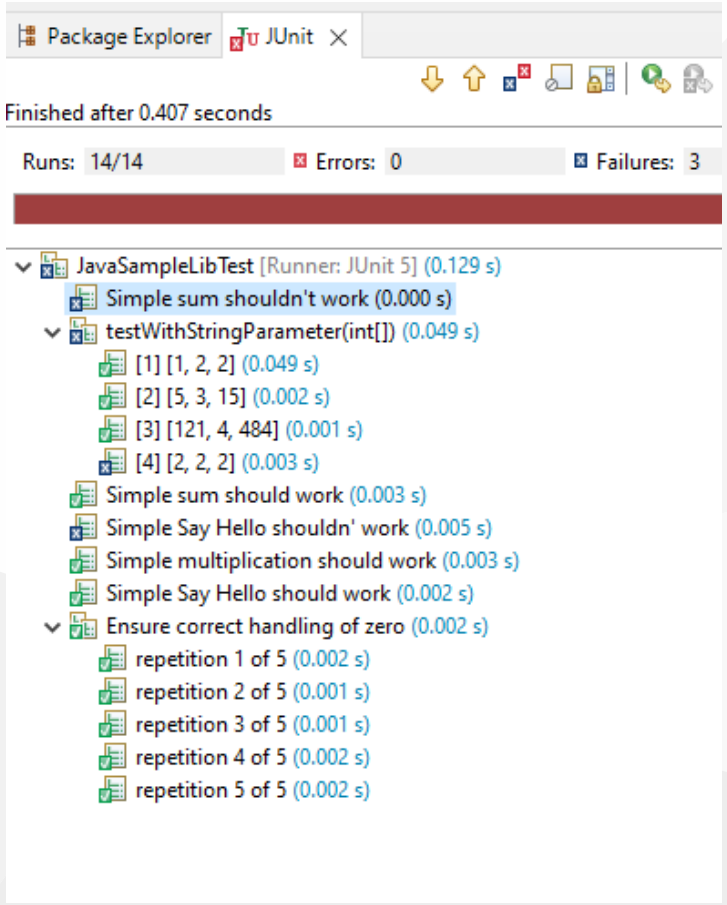
now lets copy tests from other projects

```
Convert source codes to java codes...
```

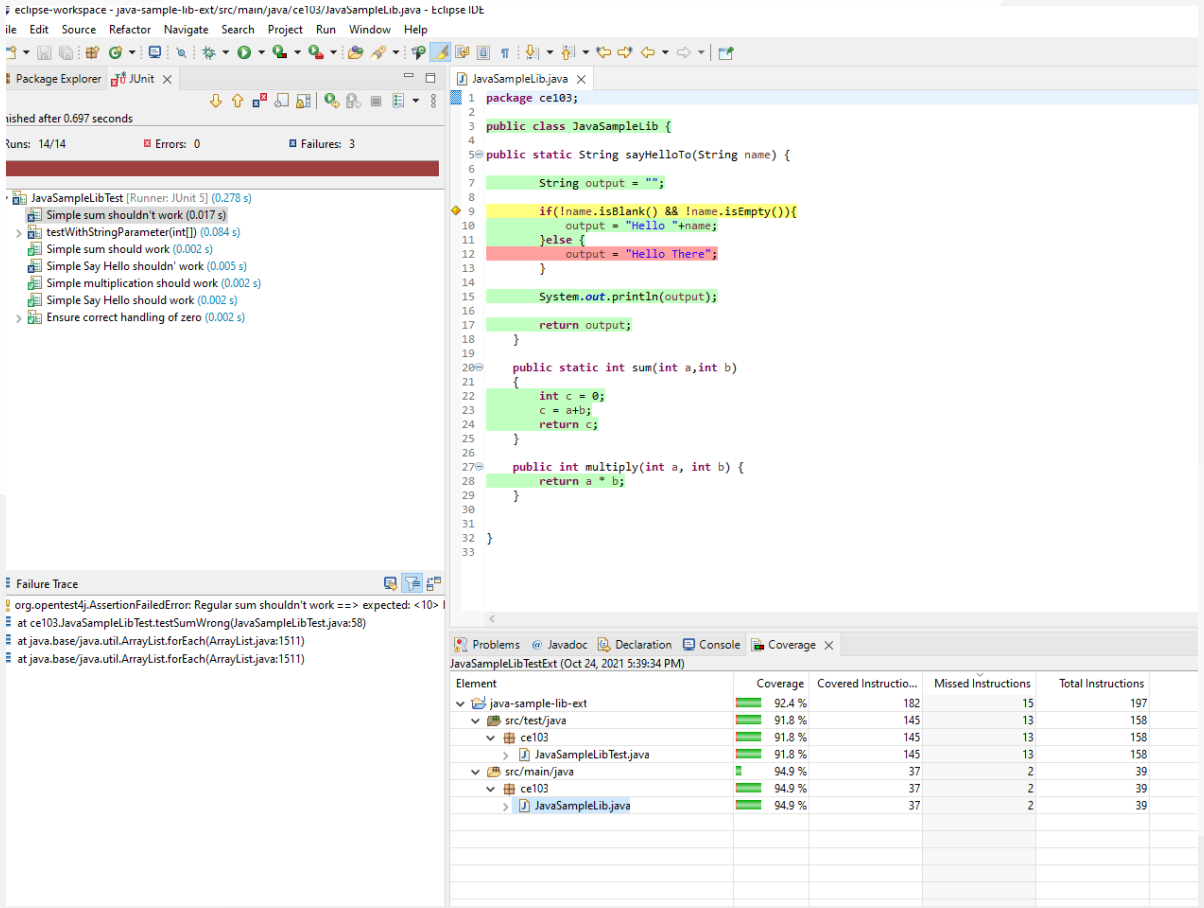
Eclipse IDE (JUnit4, JUnit5) + Java Unit Test



Eclipse IDE (JUnit4, JUnit5) + Java Unit Test



Eclipse IDE (JUnit4, JUnit5) + Java Unit Test



Eclipse IDE (JUnit4, JUnit5) + Java Unit Test

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

TDD (Test Driven Development)

- Test Driven Development (TDD)
  - https://en.wikipedia.org/wiki/Test-driven_development
- Acceptance Test Driven Development (ATDD)
  - https://en.wikipedia.org/wiki/Acceptance_test-driven_development
- Also check out
  - https://en.wikipedia.org/wiki/Kent_Beck
- Extreme Programming
  - https://en.wikipedia.org/wiki/Extreme_programming
- Software Design Patterns
  - https://en.wikipedia.org/wiki/Software_design_pattern

Test and Deployment Automation Management

There are several Continues-Integration services online as follow;

- Travis-CI
- Appveyor
- Jenkins
- CircleCI
- GitLab
- Pantheon
- GitHub
- Bitrise
- Flosum
- Buddy
- Semaphore

Test and Deployment Automation Management

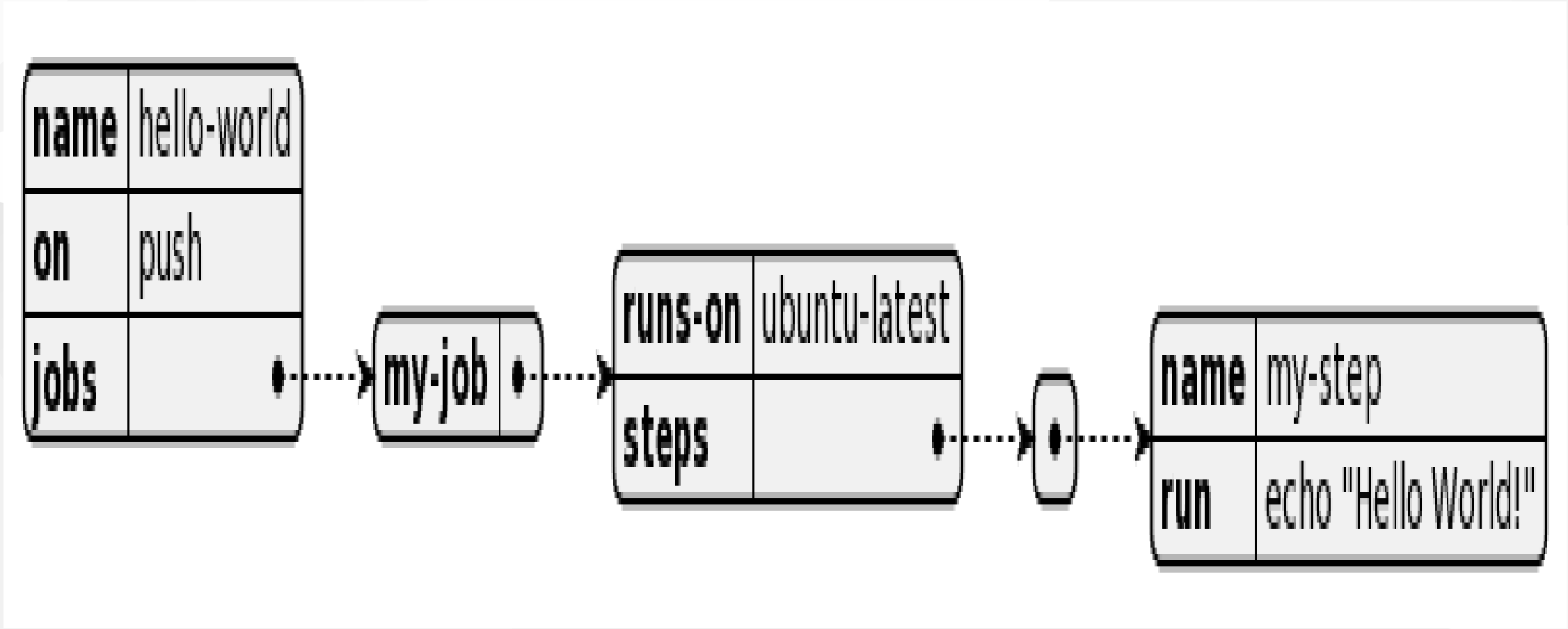
- Github provides Github Actions for Releases and Tests
- Jenkins has on promise solutions private development

Test and Deployment Automation Management

- GitHub Actions provide several actions and marketspace
  - https://github.com/marketplace/actions/build-c-project
- Also, we Can Provide Our Custom Actions

```yaml
name: hello-world
on: push
jobs:
  my-job:
    runs-on: ubuntu-latest
    steps:
      - name: my-step
        run: echo "Hello World!"
```

Test and Deployment Automation Management

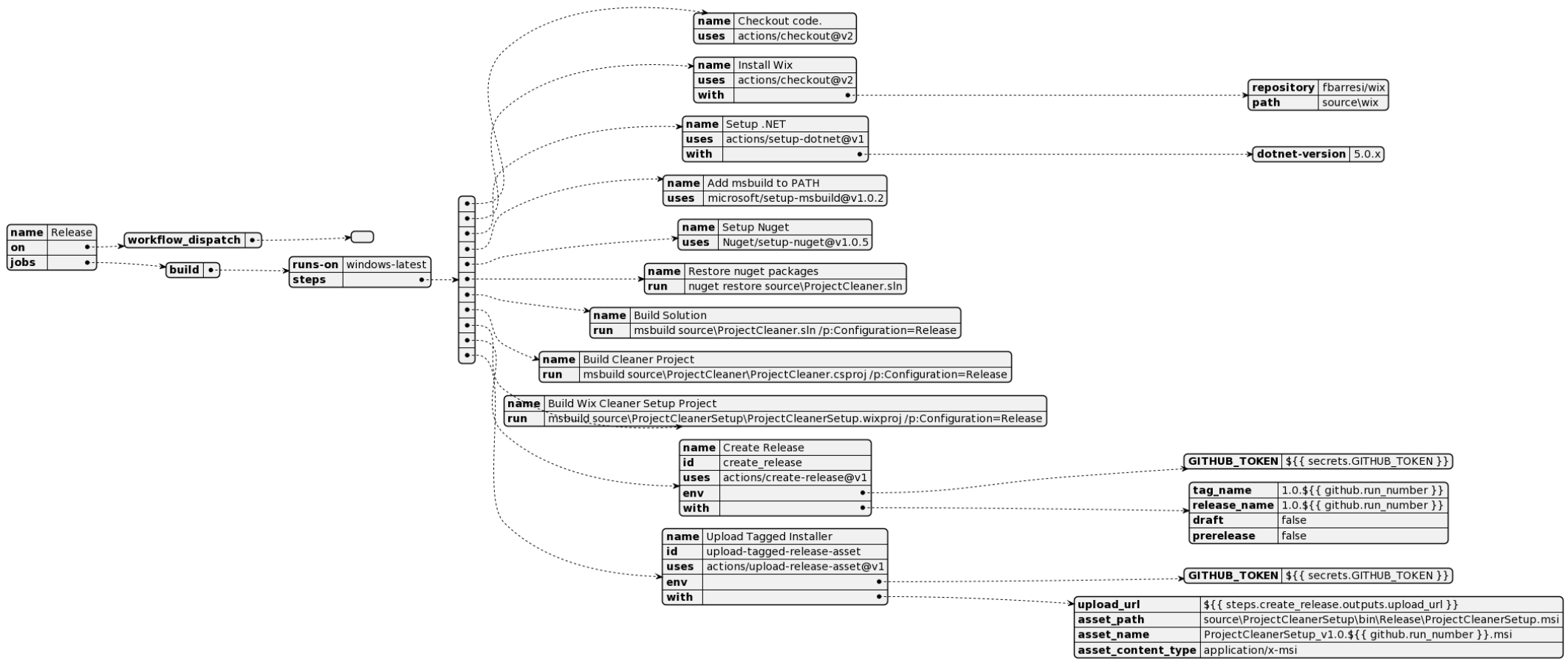


Test and Deployment Automation Management

- https://github.com/ucoruh/project-cleaner/blob/main/.github/workflows/dotnet-desktop.yml

This action build c# application and generates setup manually.

- Also there is a nice web example
  - https://dev.to/geromegrignon/github-actions-full-ci-cd-javascript-workflow-39om


name Release
on
jobs
workflow_dispatch
build
runs-on windows-latest
steps
name Checkout code.
uses actions/checkout@v2
name Install Wix
uses actions/checkout@v2
with
repository fbarresi/wix
path source\wix
name Setup .NET
uses actions/setup-dotnet@v1
with
dotnet-version 5.0.x
name Add msbuild to PATH
uses microsoft/setup-msbuild@v1.0.2
name Setup Nuget
uses Nuget/setup-nuget@v1.0.5
name Restore nuget packages
run nuget restore source\ProjectCleaner.sln
name Build Solution
run msbuild source\ProjectCleaner.sln /p:Configuration=Release
name Build Cleaner Project
run msbuild source\ProjectCleaner\ProjectCleaner.csproj /p:Configuration=Release
name Build Wix Cleaner Setup Project
run msbuild source\ProjectCleanerSetup\ProjectCleanerSetup.wixproj /p:Configuration=Release
name Create Release
id create_release
uses actions/create-release@v1
env
with
GITHUB_TOKEN ${{ secrets.GITHUB_TOKEN }}
tag_name 1.0.${{ github.run_number }}
release_name 1.0.${{ github.run_number }}
draft false
prerelease false
name Upload Tagged Installer
id upload-tagged-release-asset
uses actions/upload-release-asset@v1
env
with
GITHUB_TOKEN ${{ secrets.GITHUB_TOKEN }}
upload_url ${{ steps.create_release.outputs.upload_url }}
asset_path source\ProjectCleanerSetup\bin\Release\ProjectCleanerSetup.msi
asset_name ProjectCleanerSetup_v1.0.${{ github.run_number }}.msi
asset_content_type application/x-msi

References

GitHub - MicrosoftDocs/cpp-docs: C++ Documentation

$$End - Of - Week - 4$$