

# CE103 Algorithms and Programming I

## Week-2

### Software Development Environments

Download [DOC](#), [SLIDE](#), [PPTX](#)



## Outline

- Flowgorithm
- Introduction to Analysis of Algorithms
- Programming Environment Setup and Configuration
  - C/C++ (DevC++, Code Blocks, MinGW, LLVM, VsCode, VisualStudio, Notepad++, Vi/Vim, Eclipse, Netbeans, Cmake/Make)
  - Java (VsCode, Notepad++, Eclipse, Netbeans, Cmake)
  - C# (VsCode, Notepad++, VsCode, VisualStudio, Cmake)

## Outline

- Common Tools and Platforms
  - Fatih Kalem, Notepad++ ,HxD, MarktextApp,Cygwin,Dependency Walker,Doxygen,Sonarlint, Codepen.io, Codebeautify.org, Codeshare.io,AsciiFlow.com, Freemind, Mockflow, Wireflow, PlantUML,Drawio,Putty, MobaXterm,Teamviewer,AnyDesk,Paletton.com,Colorhunt.co,Understand,JD Project,Cutter,IDA Pro / Freeware,pythontutor,godbolt,scrcpy,Travis- CI,AppVeyor,Jenkins,Vagrant,Docker / Docker Compose / Kubernetes,Nuget,SCV Cryptomanager,Addario CryptoBench,Raymond's MD5 & SHA Checksum Utility,SlavaSoft HashCalc,Portable PGP, and more ...

# Algorithm Basics

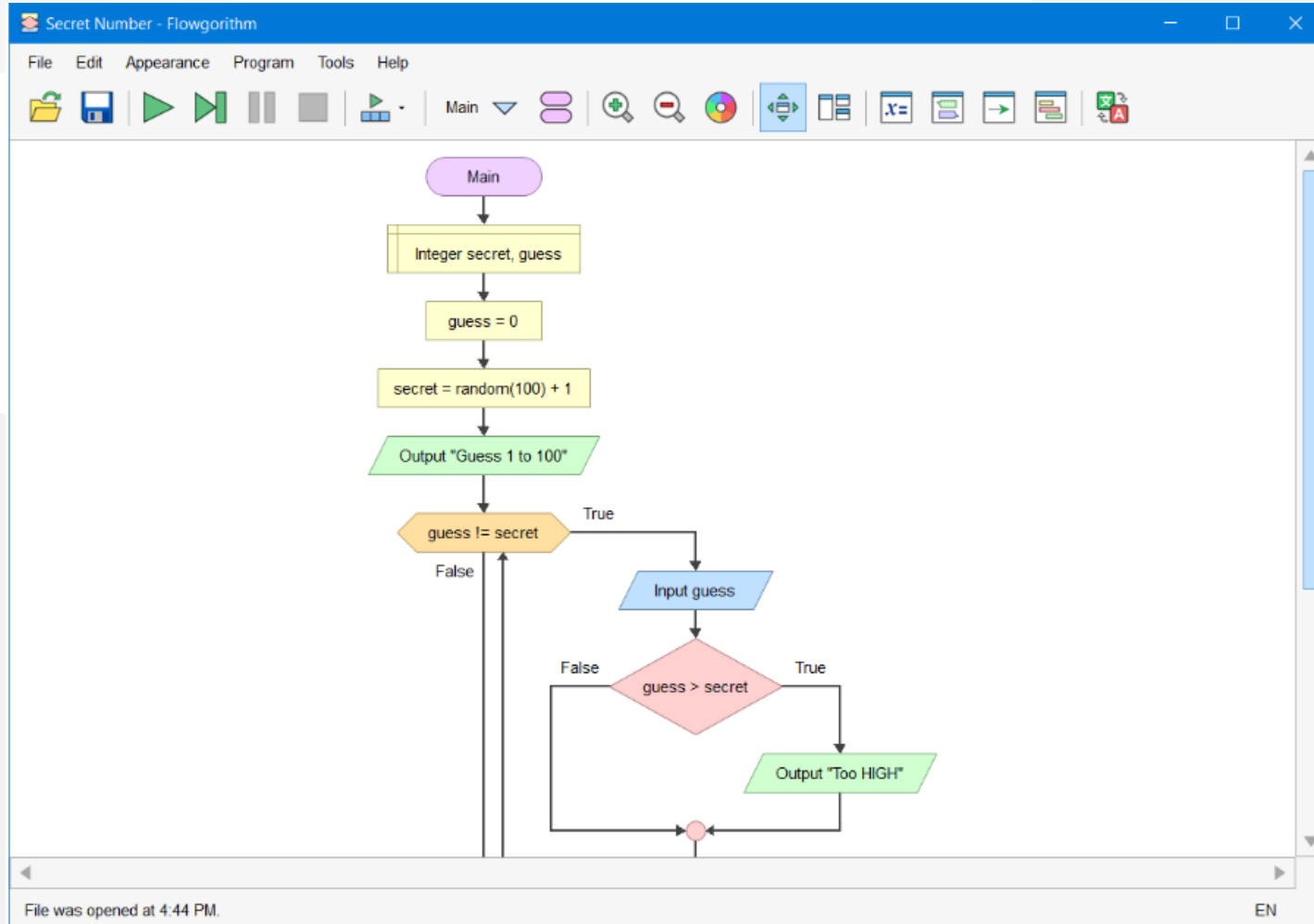


## Flowgorithm (1)

- <http://www.flowgorithm.org/>
- [Flowgorithm - Documentation](#)
- <https://github.com/timoteoponce/flowgorithm-examples>

## Flowgorithm (2)

- Main Window

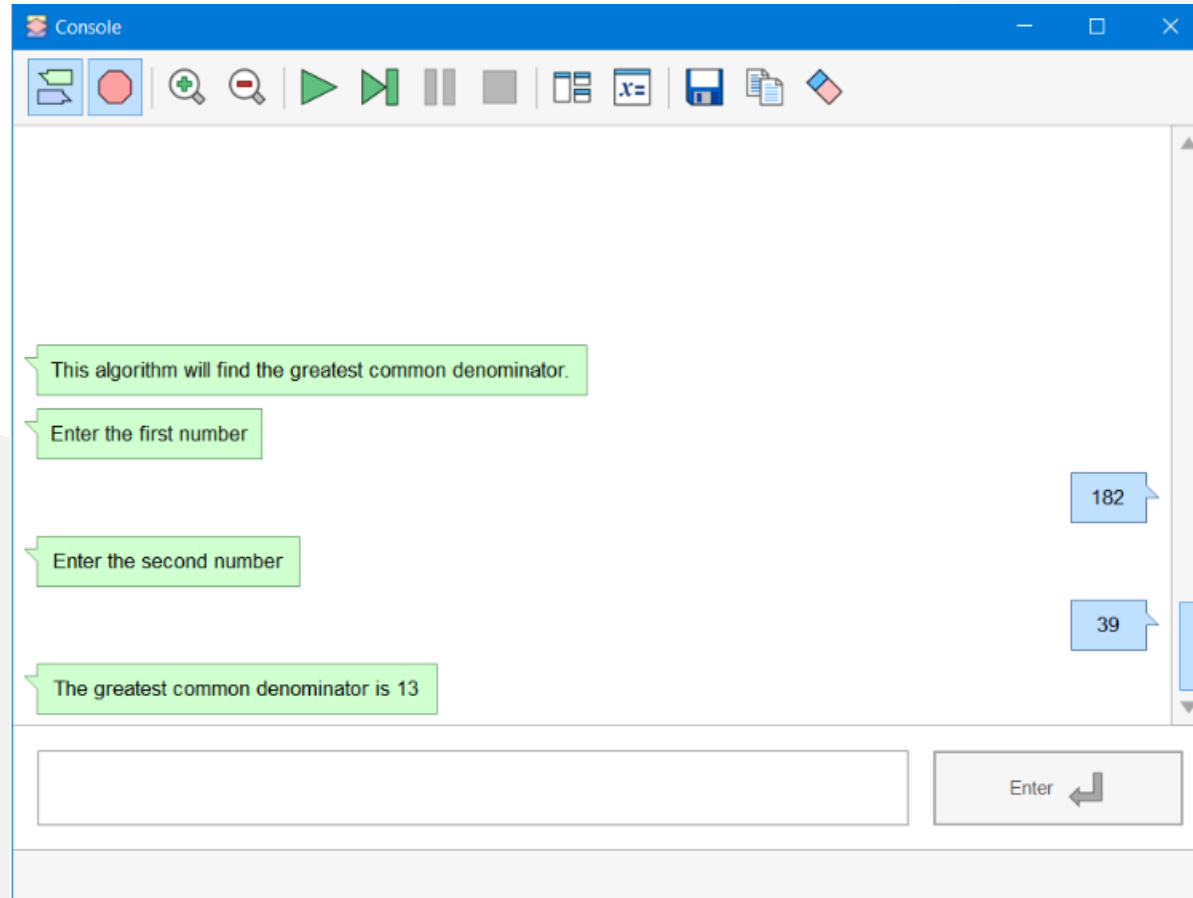


## Flowgorithm (3)

- Console Window
  - The classic method to interact with the computer is to use the "Console". Flowgorithm attempts to make it look like a typical instant messenger window. The "chat bubbles" are color coded to match the Input and Output shapes used in the flowchart. If you don't want to use the chat bubbles, you can also toggle between them and the classical plain text.

## Flowgorithm (4)

- Console Window

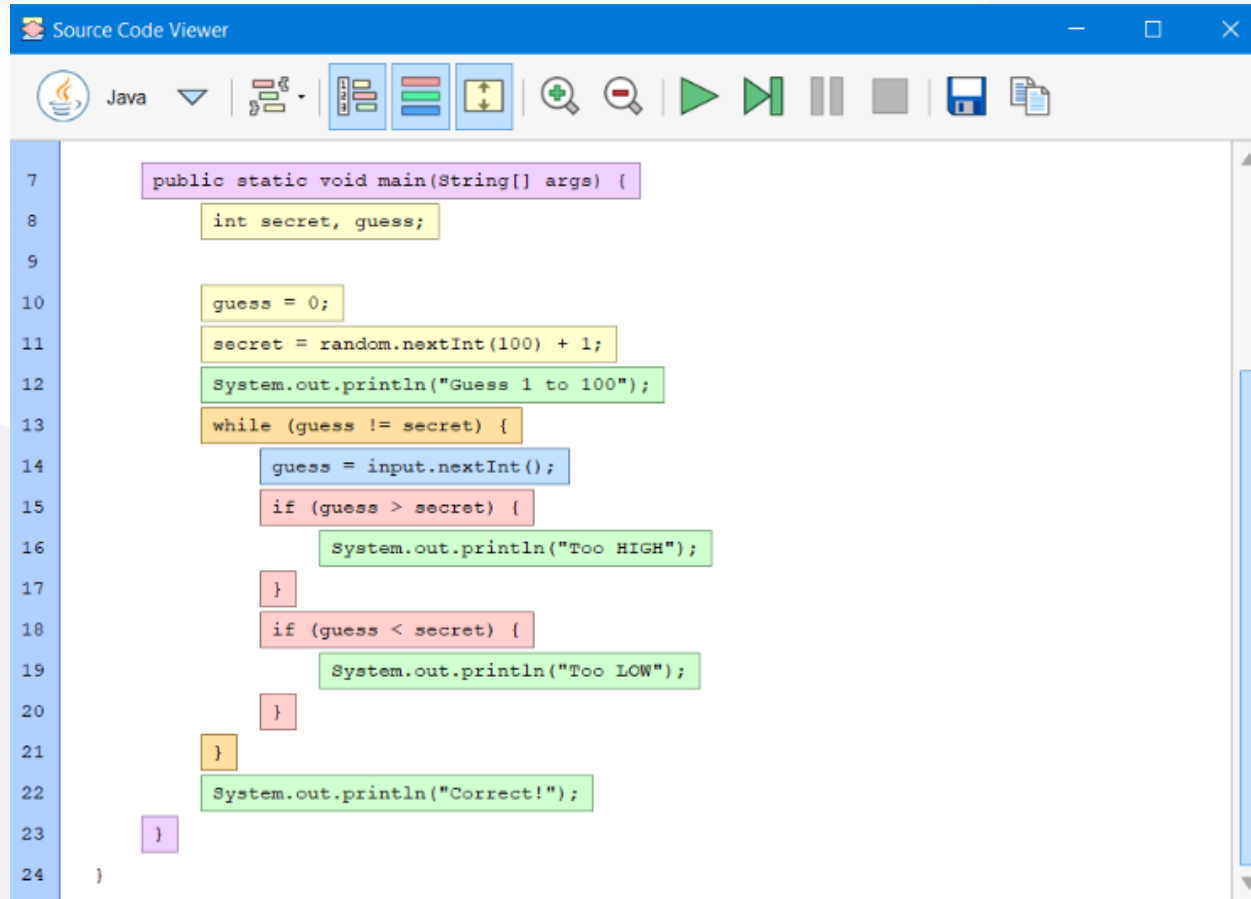


## Flowgorithm (5)

- Source Code Viewer Window
  - The Source Code Viewer can convert your flowchart to several major programming languages. So, if you planning to learn a high-level language, then this feature should help you along the way.

## Flowgorithm (6)

- Source Code Viewer Window



The screenshot shows a window titled "Source Code Viewer" with a blue title bar. The window contains a Java code editor with a toolbar at the top. The toolbar includes icons for Java, file operations, and execution. The code is as follows:

```
7     public static void main(String[] args) {  
8         int secret, guess;  
9  
10        guess = 0;  
11        secret = random.nextInt(100) + 1;  
12        System.out.println("Guess 1 to 100");  
13        while (guess != secret) {  
14            guess = input.nextInt();  
15            if (guess > secret) {  
16                System.out.println("Too HIGH");  
17            }  
18            if (guess < secret) {  
19                System.out.println("Too LOW");  
20            }  
21        }  
22        System.out.println("Correct!");  
23    }  
24 }
```

## Flowgorithm (7)

- Variable Watch Window
  - The variable watch window is used to keep track of how your variables are changing as your program executes. Each variable is color-coded based on its data type. At a glance, you can tell exactly what type of data is being stored - and catch where you may want to use a different data type.

## Flowgorithm (8)

- Variable Watch Window

The image displays two screenshots of the Variable Watch window in Flowgorithm, showing the state of variables during program execution.

**Left Screenshot:**

- Main:** (Blue bar)
- year:** (Blue bar) 1947
- ratio:** (Purple bar) 1.618
- name:** (Red bar) Sacramento State
- done:** (Green bar) true

**Right Screenshot:**

- Main:** (Blue bar)
- year:** (Blue bar) 1947
- squares:** (Blue bar) Array of values:

0	0
1	1
2	4
3	9
4	16
5	25
6	36
7	49
- ratio:** (Purple bar) 1.618



## Flowgorithm (8)

- More Resources for Flowgorithm
  - [Flowgorithm Tutorial - TestingDocs.com](https://www.testingdocs.com/flowgorithm-tutorial/)

## Pseudocode (1)

- Algorithm design language
  - [Pseudocode - Wikipedia](#)
  - [Pseudocode Examples](#)
  - [How to write a Pseudo Code? - GeeksforGeeks](#)

## Introduction to Analysis of Algorithms

- In this course, we will learn how to code with several development environments and next term we will see an analysis of algorithms in detail.
- This topic is covered in the following link:
  - [CE100 Introduction to Analysis of Algorithms](#)

## Programming Environment Setup and Configuration

- Programming life is not about only learning how to code. Mostly you need to use several code development environments and you need to learn how to use them efficiently.

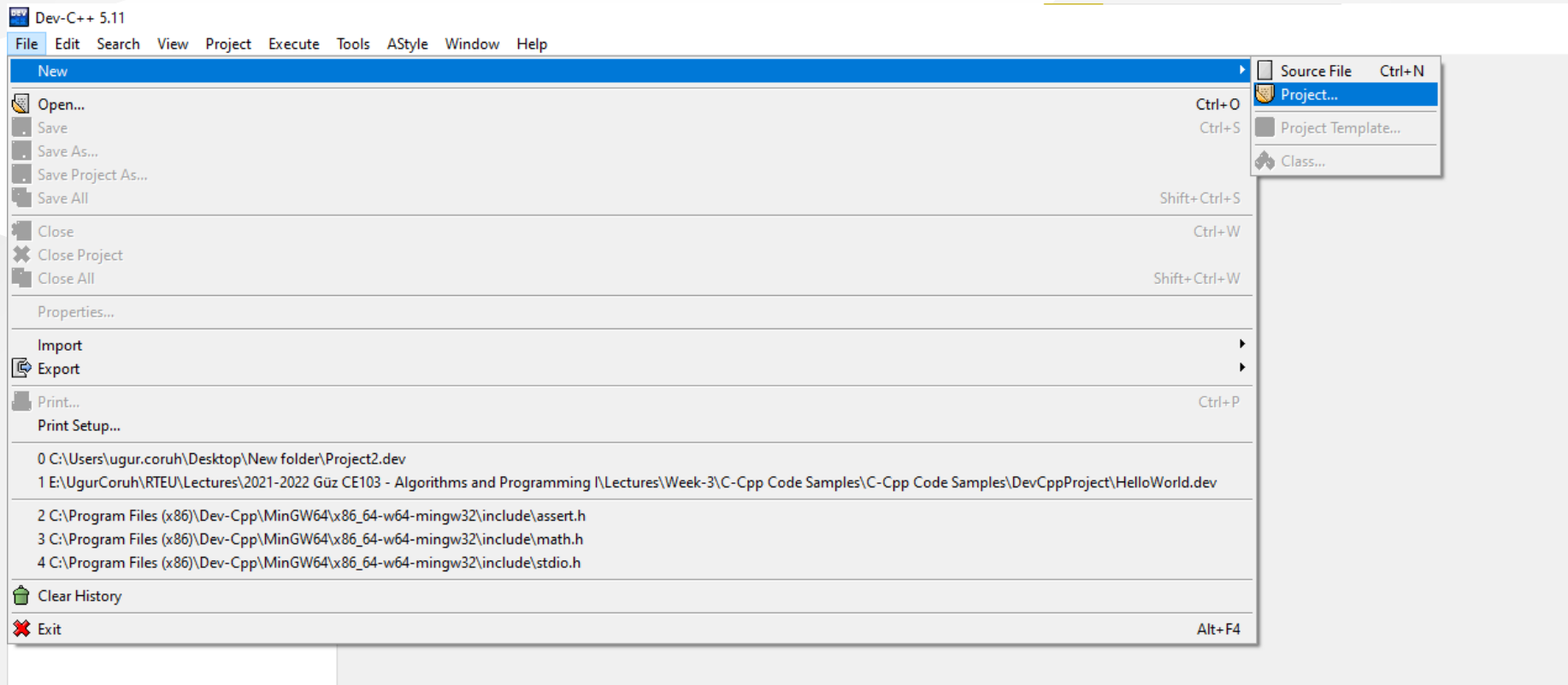
# C / C++ Environment and Development

## DevCpp (Install / Compile / Run / Debug) (1)

- Download DevC++ IDE from the following link
  - <https://www.bloodshed.net/>

## DevCpp (Install / Compile / Run / Debug) (2)

- Open DevC++ IDE for C Project Generation **Open File->New->Project**



### DevCpp (Install / Compile / Run / Debug) (3)

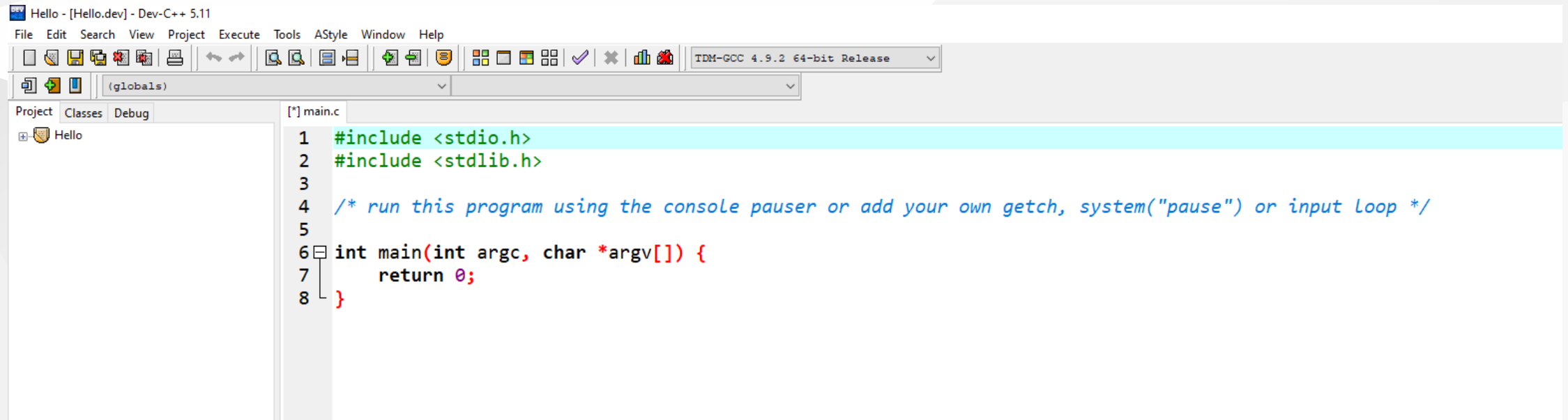
Select **Console Application** from **Basic** tab and with **C Project** Option and write a project name such as "**Hello**" then press OK

Select a folder and save **Hello.dev** project file.



## DevCpp (Install / Compile / Run / Debug) (4)

- You will see a sample main with an empty body



The screenshot shows the Dev-C++ IDE interface. The title bar reads "Hello - [Hello.dev] - Dev-C++ 5.11". The menu bar includes File, Edit, Search, View, Project, Execute, Tools, AStyle, Window, and Help. The toolbar contains various icons for file operations and execution. The compiler is identified as "TDM-GCC 4.9.2 64-bit Release". The project explorer on the left shows a project named "Hello". The main editor window displays the following code in "main.c":

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 /* run this program using the console pauser or add your own getch, system("pause") or input loop */
5
6 int main(int argc, char *argv[]) {
7     return 0;
8 }
```

## DevCpp (Install / Compile / Run / Debug) (5)

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

/* run this program using the console pauser or add your own getch, system("pause") or input loop */
int main(int argc, char *argv[]) {
    retAdd 0;
}
```

## DevCpp (Install / Compile / Run / Debug) (6)

- Add the following line in the main function. This will write "Hello, World!" on the screen and then wait for a keypress to exit from the application

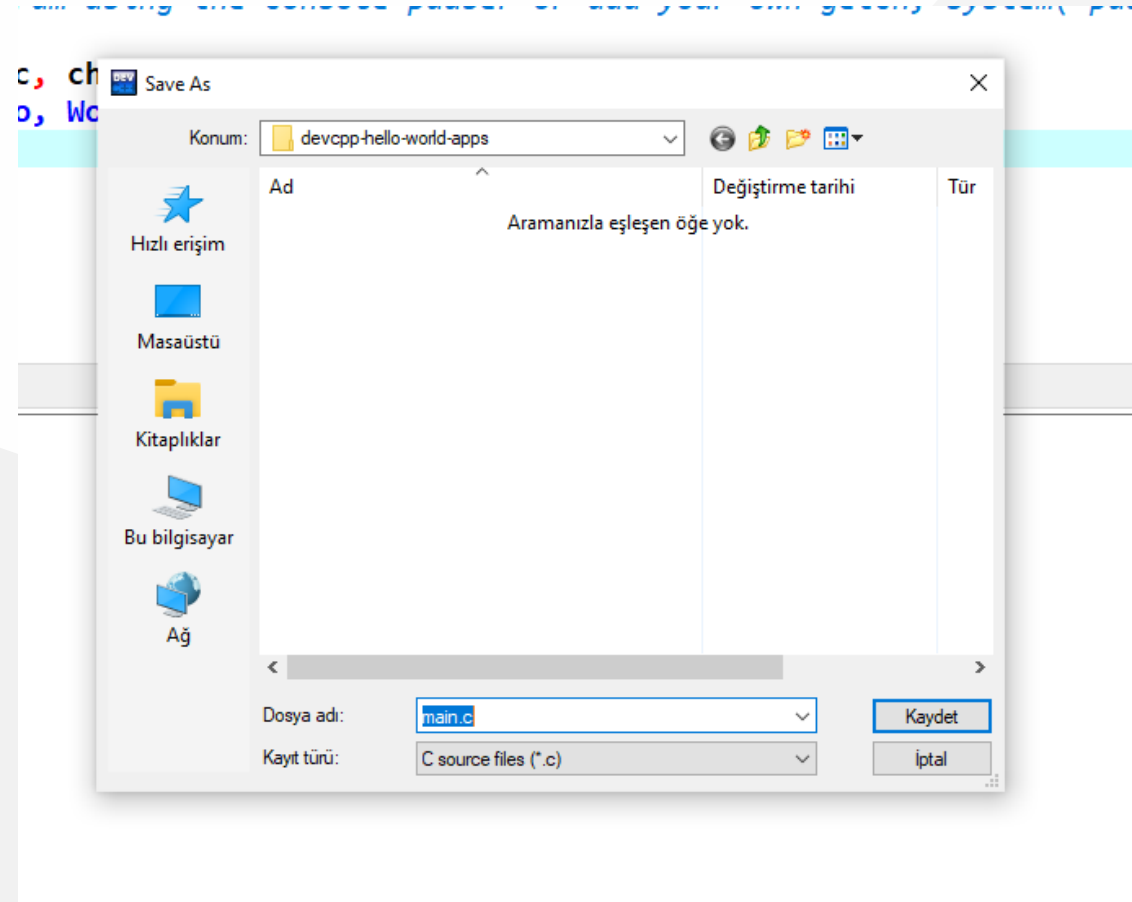
```
#include <stdio.h>
#include <stdlib.h>

/* run this program using the console pauser or add your own getch, system("pause") or input loop */

int main(int argc, char *argv[]) {
    printf("Hello, World!");
    getch();
    return 0;
}
```

## DevCpp (Install / Compile / Run / Debug) (7)

- Then save the file



## DevCpp (Install / Compile / Run / Debug) (8)

- Use from menu **Execute->Compile F5** to generate **Hello.exe**

The screenshot shows the Dev-C++ IDE interface. The 'Execute' menu is open, and the 'Compile F5' option is selected. The code editor displays the following C++ code:

```

#include <stdio.h>
#include <stdlib.h>

// * program using the console pauser or add your own getch, system("pause") or input loop */

int main(int argc, char *argv[]) {
    printf("Hello, World!");
    return 0;
}

```

The console window shows the compilation process:

```

Compiling project changes...
-----
- Project Filename: E:\UgurCoruh\RTEU\Lectures\2021-2022 Güz CE103 - Algorithms and Programming I\Lectures\ce103-algorithms-and-programming-I\Week-2\devcpp-hello-world-apps\Hello.dev
- Compiler Name: TDM-GCC 4.9.2 64-bit Release
-----
Building makefile...
-----
- Filename: E:\UgurCoruh\RTEU\Lectures\2021-2022 Güz CE103 - Algorithms and Programming I\Lectures\ce103-algorithms-and-programming-I\Week-2\devcpp-hello-world-apps\Makefile.win
-----
Processing makefile...
-----
- Makefile Processor: C:\Program Files (x86)\Dev-Cpp\MinGW64\bin\mingw32-make.exe
- Command: mingw32-make.exe -f "E:\UgurCoruh\RTEU\Lectures\2021-2022 Güz CE103 - Algorithms and Programming I\Lectures\ce103-algorithms-and-programming-I\Week-2\devcpp-hello-world-apps\Makefile.win" all

gcc.exe -c main.c -o main.o -I"C:/Program Files (x86)/Dev-Cpp/MinGW64/include" -I"C:/Program Files (x86)/Dev-Cpp/MinGW64/x86_64-w64-mingw32/include" -I"C:/Program Files (x86)/Dev-Cpp/MinGW64/lib/gcc/x86_64-w64-gcc.exe main.o -o Hello.exe -L"C:/Program Files (x86)/Dev-Cpp/MinGW64/lib" -L"C:/Program Files (x86)/Dev-Cpp/MinGW64/x86_64-w64-mingw32/lib" -static-libgcc

Compilation results...
-----
- Errors: 0
- Warnings: 0
- Output Filename: E:\UgurCoruh\RTEU\Lectures\2021-2022 Güz CE103 - Algorithms and Programming I\Lectures\ce103-algorithms-and-programming-I\Week-2\devcpp-hello-world-apps\Hello.exe
- Output Size: 128,103515625 KiB
- Compilation Time: 2,13s

```

## DevCpp (Install / Compile / Run / Debug) (9)

- You can find the generated `Hello.exe` path from `Compile.log` as follow. Check the Output Filename

```

Compiling project changes...
-----
- Project Filename: E:\UgurCoruh\RTEU\Lectures\2021-2022 Güz CE103 - Algorithms and Programming I\Lectures\ce103-algorithms-and-programming-I\Week-2\devcpp-hello-world-apps\Hello.dev
- Compiler Name: TDM-GCC 4.9.2 64-bit Release

Building makefile...
-----
- Filename: E:\UgurCoruh\RTEU\Lectures\2021-2022 Güz CE103 - Algorithms and Programming I\Lectures\ce103-algorithms-and-programming-I\Week-2\devcpp-hello-world-apps\Makefile.win

Processing makefile...
-----
- Makefile Processor: C:\Program Files (x86)\Dev-Cpp\MinGW64\bin\mingw32-make.exe
- Command: mingw32-make.exe -f "E:\UgurCoruh\RTEU\Lectures\2021-2022 Güz CE103 - Algorithms and Programming I\Lectures\ce103-algorithms-and-programming-I\Week-2\devcpp-hello-world-apps\Makefile.win" all

gcc.exe -c main.c -o main.o -I"C:/Program Files (x86)/Dev-Cpp/MinGW64/include" -I"C:/Program Files (x86)/Dev-Cpp/MinGW64/x86_64-w64-mingw32/include" -I"C:/Program Files (x86)/Dev-Cpp/MinGW64/lib/gcc/x86_64-w64-mingw32/4.9.2/include"

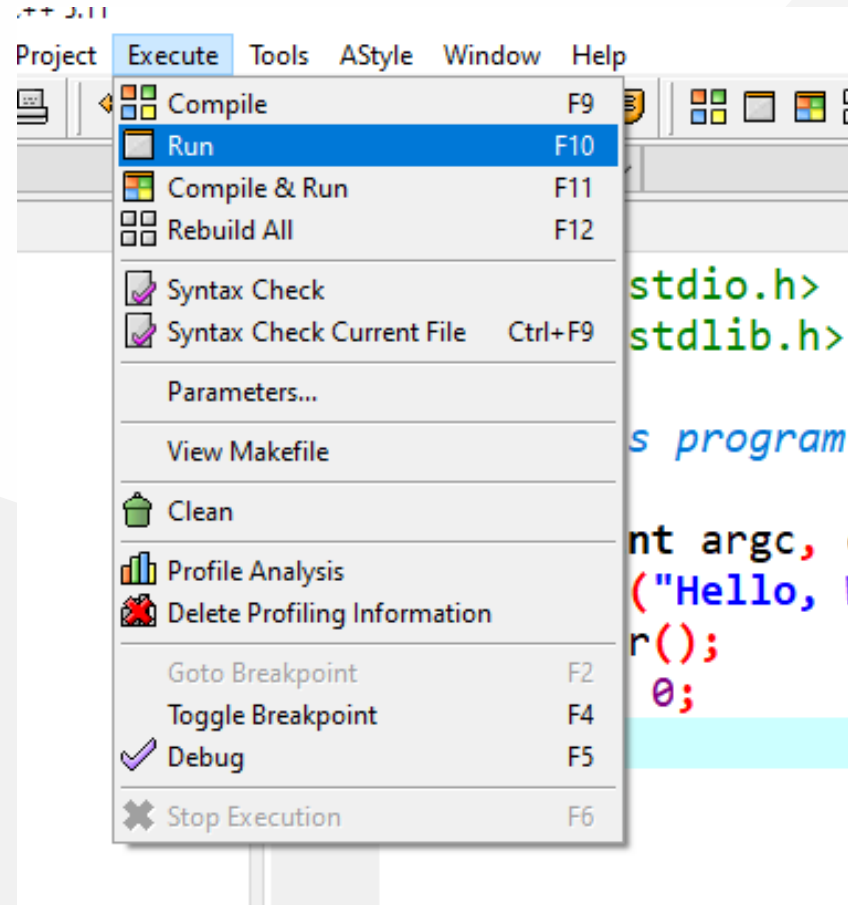
gcc.exe main.o -o Hello.exe -L"C:/Program Files (x86)/Dev-Cpp/MinGW64/lib" -L"C:/Program Files (x86)/Dev-Cpp/MinGW64/x86_64-w64-mingw32/lib" -static-libgcc

Compilation results...
-----
- Errors: 0
- Warnings: 0
- Output Filename: E:\UgurCoruh\RTEU\Lectures\2021-2022 Güz CE103 - Algorithms and Programming I\Lectures\ce103-algorithms-and-programming-I\Week-2\devcpp-hello-world-apps\Hello.exe
- Output Size: 128,103515625 KiB
- Compilation Time: 2,13s

```

## DevCpp (Install / Compile / Run / Debug) (10)

- Then you can run with `Execute->Run F10` or Directly `Compile&Run F11`



## DevC++ (Install / Compile / Run / Debug) (11)

for debugging operations, just change the code and add more statements as follow

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

/* run this program using the console pauser or add your getch, system(",pause") or input loop */

int main(int argc, char *argv[]) {

    printf("Hello, World! Step-1\n");
    printf("Hello, World! Step-2\n");
    printf("Hello, World! Step-3\n");
    printf("Hello, World! Step-4\n");
    printf("Hello, World! Step-5\n");
    printf("Hello, World! Step-6\n");

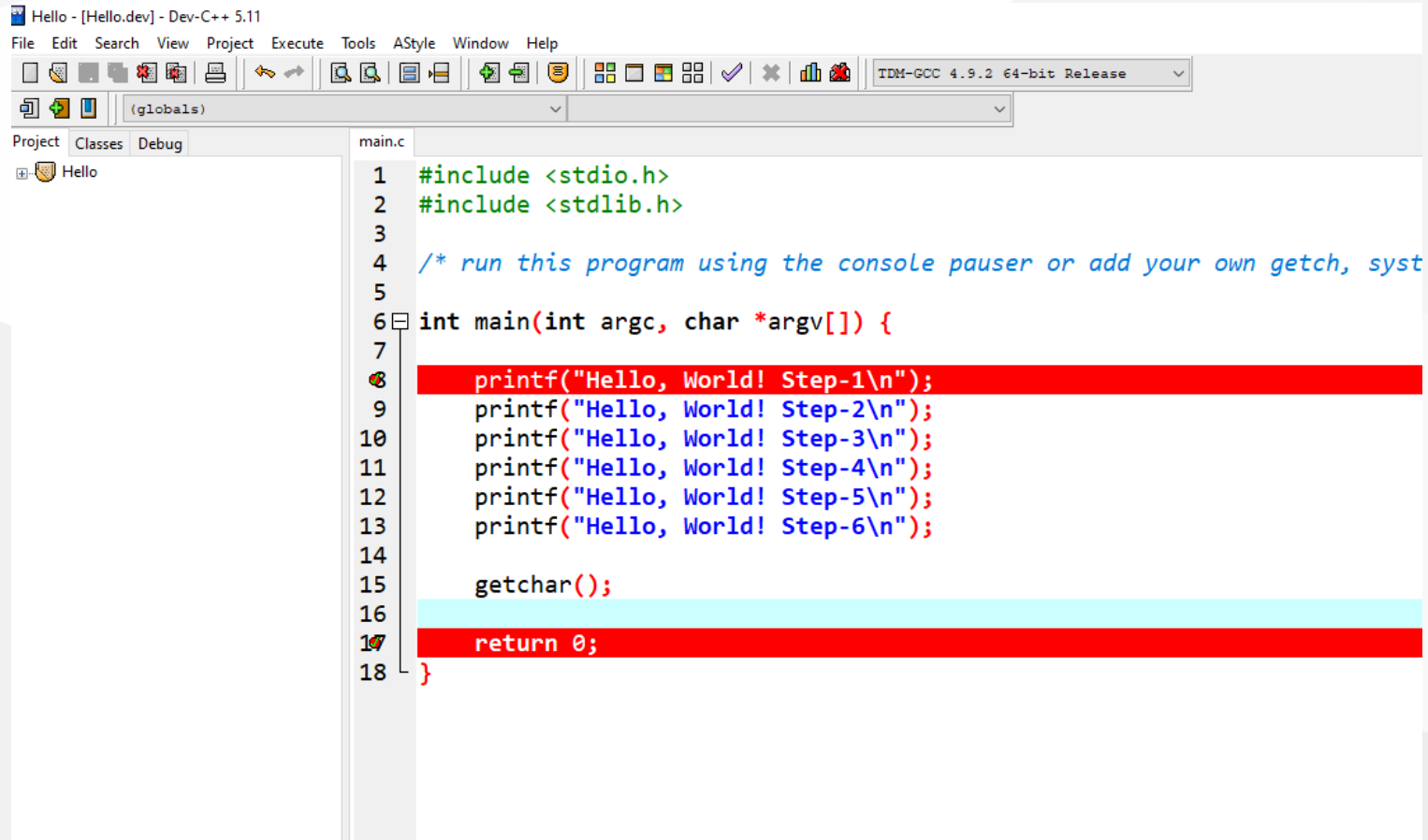
    getchar();

    return 0;
}
```



## DevCpp (Install / Compile / Run / Debug) (12)

Click on line numbers and add breakpoints for the debugger. This red point will be debugger stop points



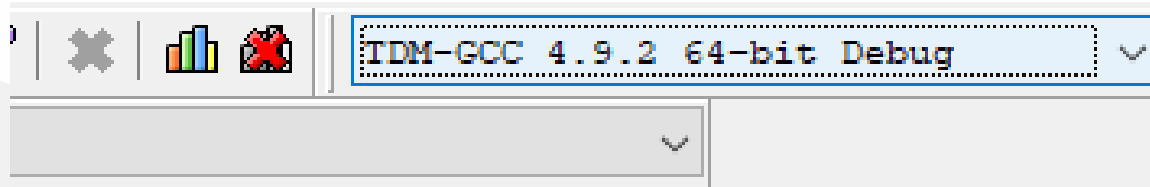
The screenshot shows the Dev-C++ IDE interface. The main window displays a C program named 'main.c'. The code is as follows:

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 /* run this program using the console pauser or add your own getch, syst
5
6 int main(int argc, char *argv[]) {
7
8     printf("Hello, World! Step-1\n");
9     printf("Hello, World! Step-2\n");
10    printf("Hello, World! Step-3\n");
11    printf("Hello, World! Step-4\n");
12    printf("Hello, World! Step-5\n");
13    printf("Hello, World! Step-6\n");
14
15    getchar();
16
17    return 0;
18 }
```

Breakpoints are indicated by red circles on the left margin of the code editor, positioned next to lines 8 and 17. The IDE also shows a menu bar (File, Edit, Search, View, Project, Execute, Tools, AStyle, Window, Help), a toolbar, and a project explorer on the left showing a project named 'Hello'.

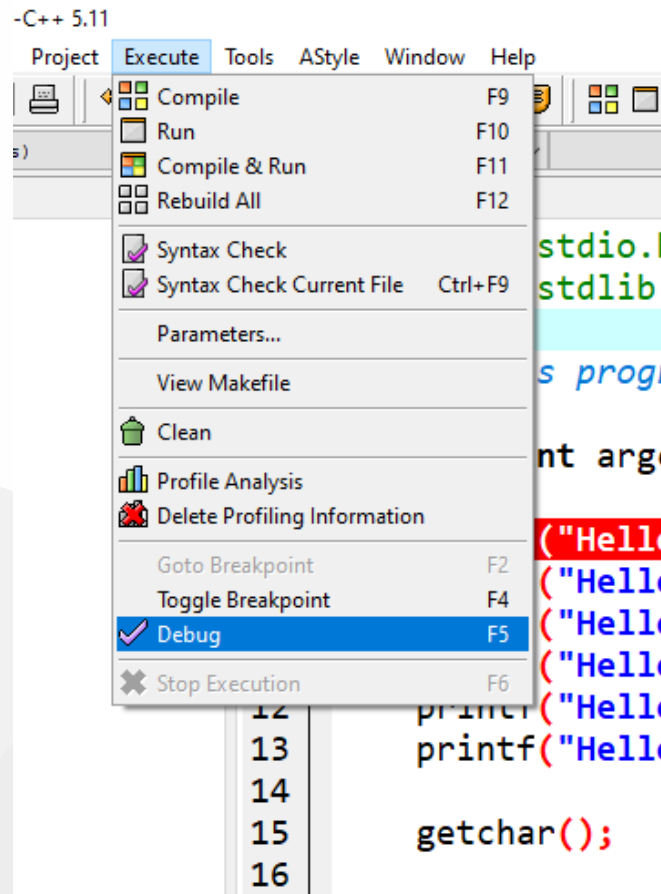
## DevCpp (Install / Compile / Run / Debug) (13)

- In the menu section, select the compiler with debug option



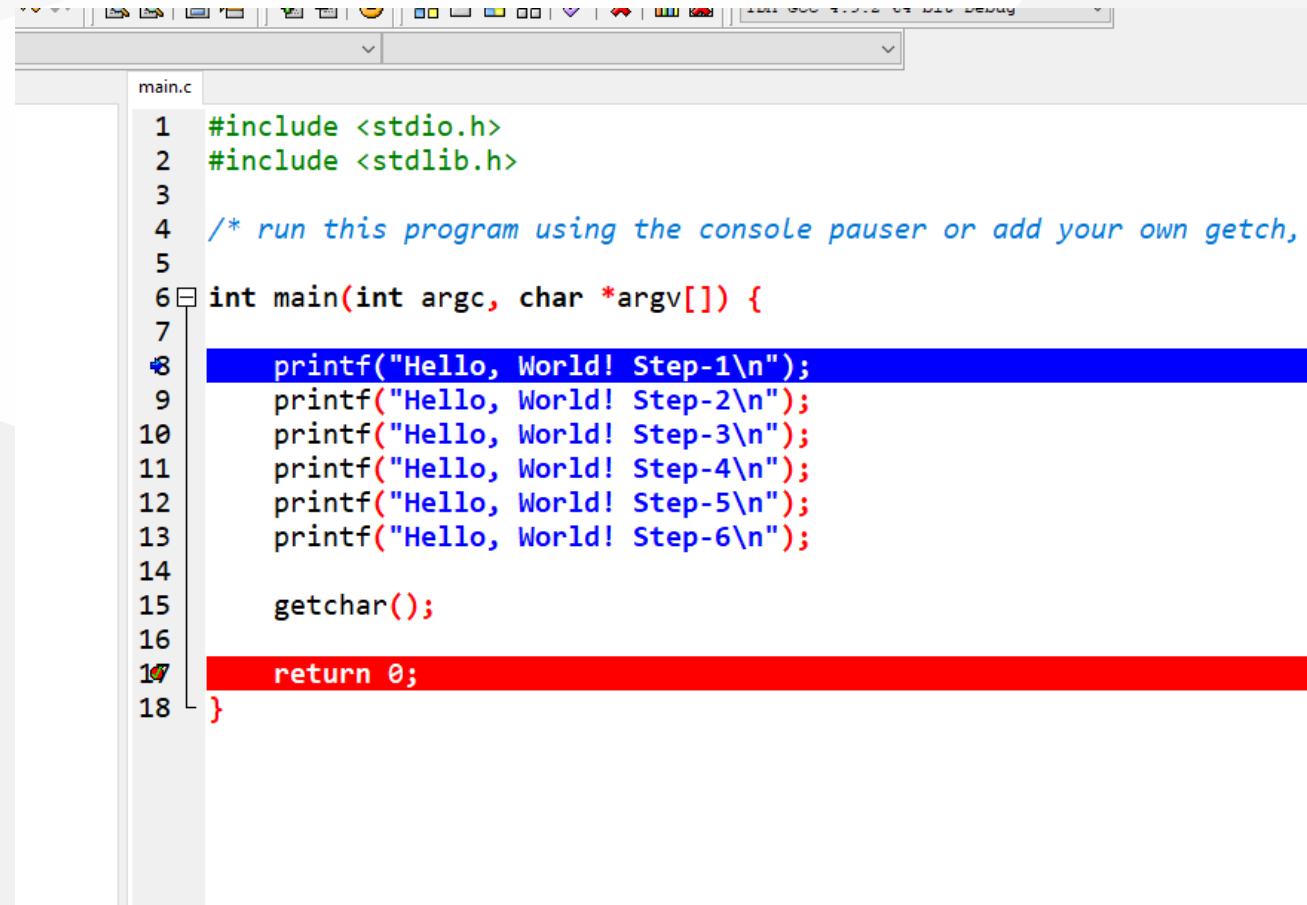
## DevCpp (Install / Compile / Run / Debug) (14)

- Compile application with debugging setting and in Execute Section use Debug F5 to start debugging



## DevCpp (Install / Compile / Run / Debug) (15)

- The debugger will stop at the breakpoint at the debug point (blue line)

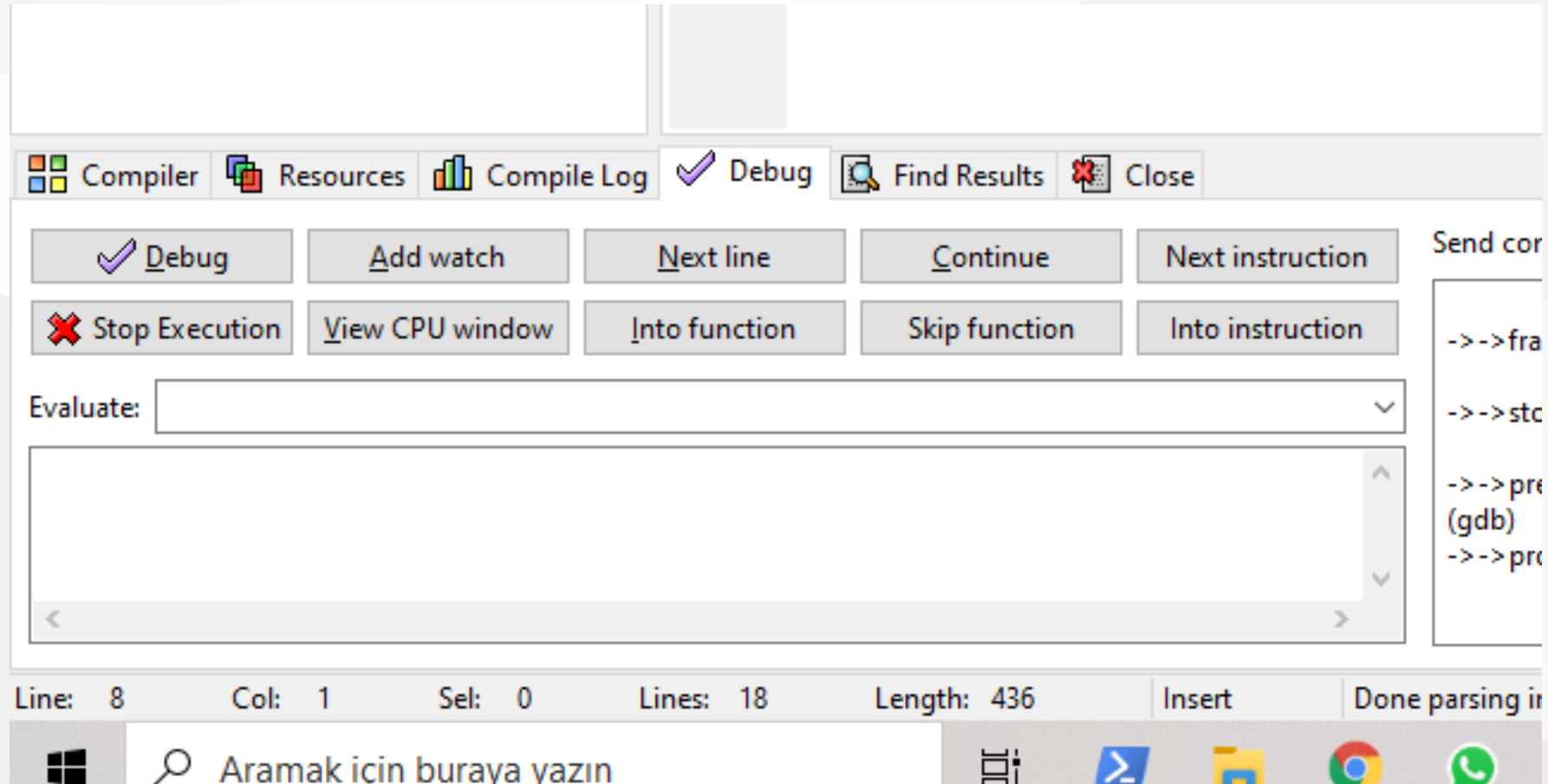
A screenshot of the Dev-Cpp IDE showing a C program named 'main.c'. The code is as follows:

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 /* run this program using the console pauser or add your own getch,
5
6 int main(int argc, char *argv[]) {
7
8 printf("Hello, World! Step-1\n");
9 printf("Hello, World! Step-2\n");
10 printf("Hello, World! Step-3\n");
11 printf("Hello, World! Step-4\n");
12 printf("Hello, World! Step-5\n");
13 printf("Hello, World! Step-6\n");
14
15 getchar();
16
17 return 0;
18 }
```

The line containing the first printf statement (line 8) is highlighted in blue, indicating a breakpoint. The line containing the return statement (line 17) is highlighted in red. The IDE interface includes a toolbar at the top and a file explorer on the left.

## DevCpp (Install / Compile / Run / Debug) (16)

- Moving to the next statement can be done via control buttons or shortcuts



## DevCpp (Install / Compile / Run / Debug) (17)

-Press `F8` to step-by-step continue

- Then go to `Project Options -> Compiler -> Linker` and set Generate debugging information to "yes", and make sure you are not using any optimization options (they're not good for debug mode). Also, check the Parameters tab, and make sure you don't have any optimization options (like `-O2` or `-O3`, but `-O0` is ok because it means no optimization) or strip option (`-s`).

## DevCpp (Install / Compile / Run / Debug) (18)

- After that, do a full rebuild ( `Ctrl-F11` ), then set a breakpoint(s) where you want the debugger to stop (otherwise it will just run the program). To set a breakpoint on a line, just click on the gutter (the gray band on the left), or press `Ctrl-F5` .

## DevCpp (Install / Compile / Run / Debug) (19)

- Now you are ready to launch the debugger, by pressing F8 or clicking the debug button. If everything goes well, the program will start, and then stop at the first breakpoint. Then you can step through the code, entering function calls, by pressing `Shift-F7` or the "step into" button, or stepping over the function calls, by pressing `F7` or the "next step" button. You can press `Ctrl-F7` or the "continue" button to continue execution till the next breakpoint. At any time, you can add or remove breakpoints.



## DevCpp (Install / Compile / Run / Debug) (20)

When the program stopped at a breakpoint and you are stepping through the code, you can display the values of various variables in your program by putting your mouse over them, or you can display variables and expressions by pressing **F4** or the "add watch" button and typing the expression.

## DevCpp (Install / Compile / Run / Debug) (21)

How do I debug using Dev-C++

## Code Blocks (Install / Compile / Run / Debug) (1)

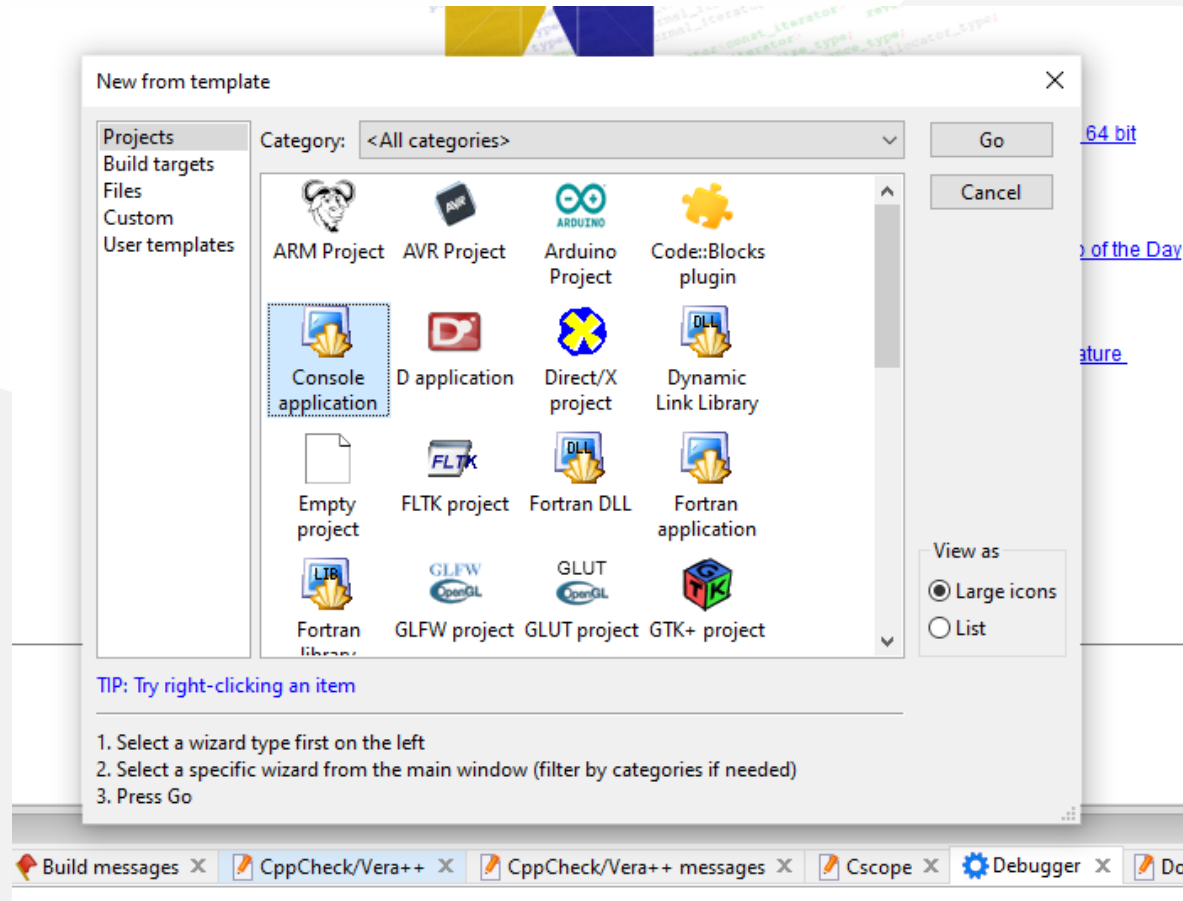
Download Code Blocks from the following link

[Binary releases - Code::Blocks](#)

## Code Blocks (Install / Compile / Run / Debug) (2)

Open Code Blocks and

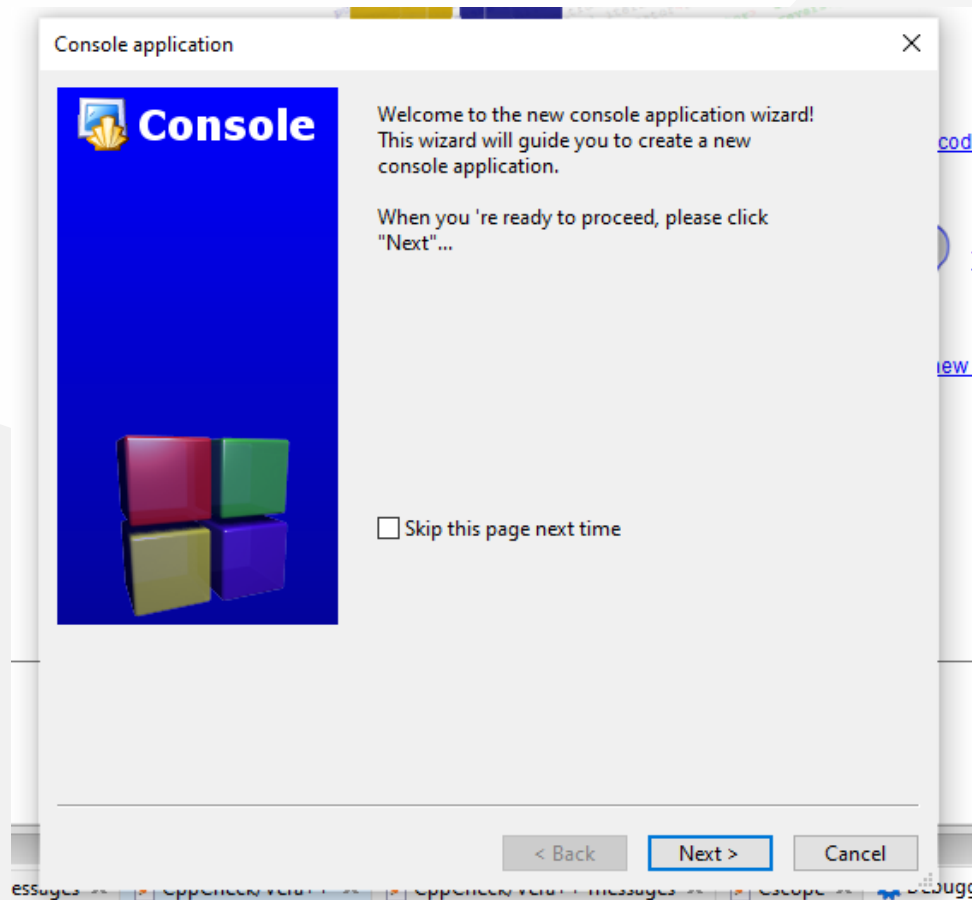
Select **File->New->Project**



## Code Blocks (Install / Compile / Run / Debug) (3)

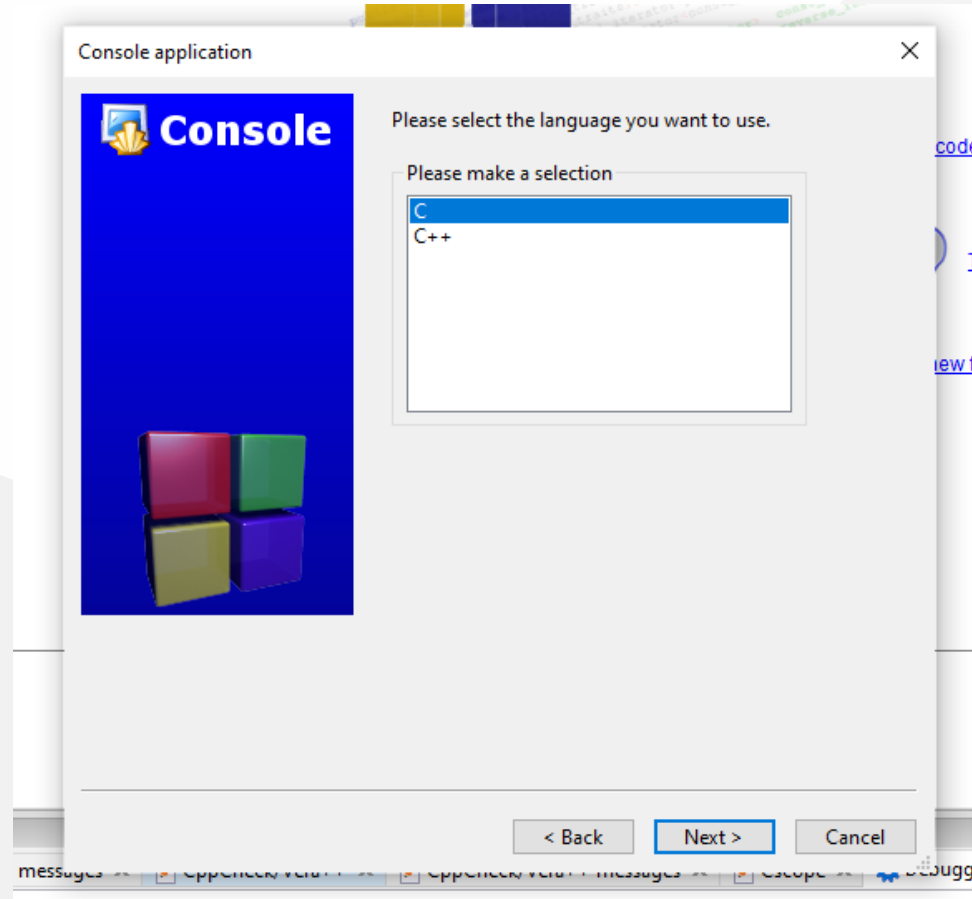
Select **Console Application**

Click **Next** from Opening Window



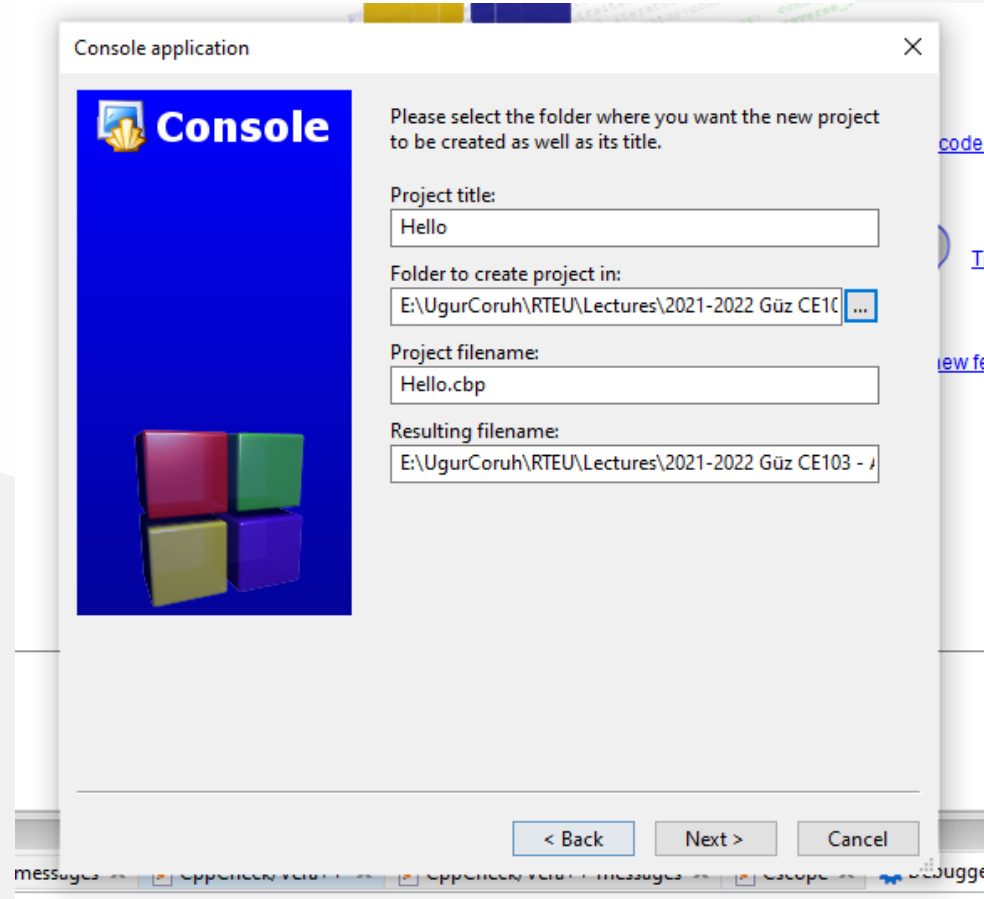
## Code Blocks (Install / Compile / Run / Debug) (4)

Select **c** for Sample Project



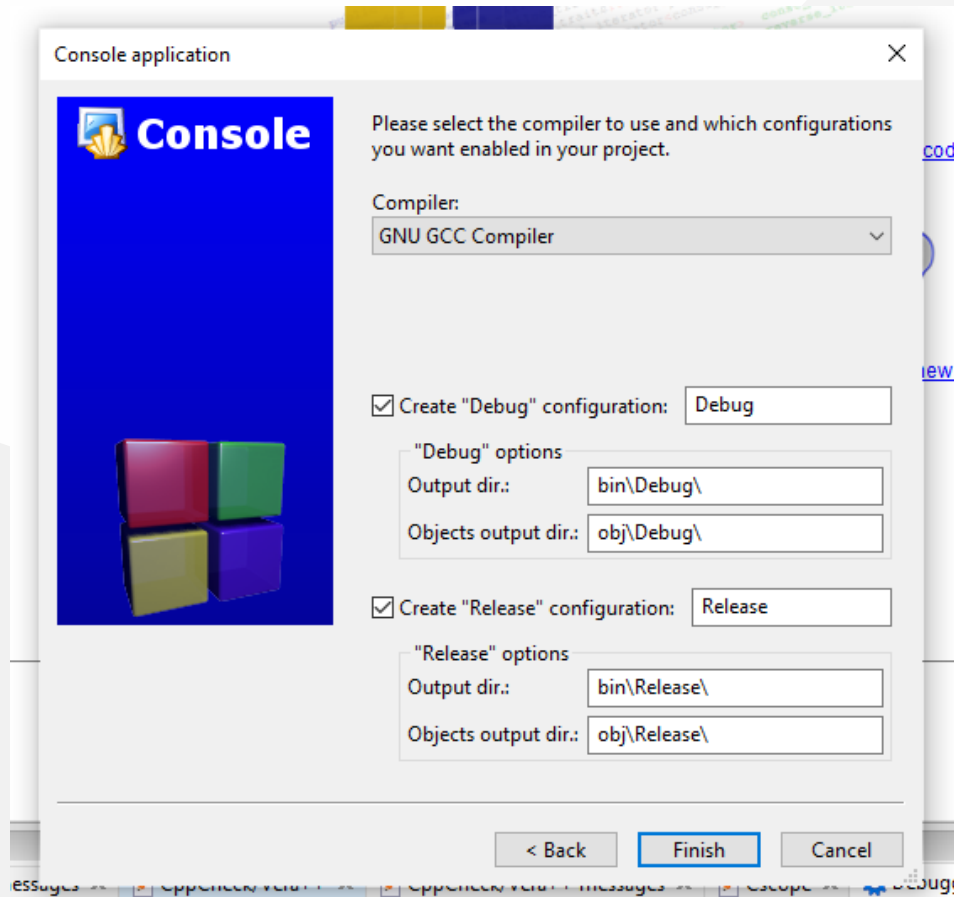
## Code Blocks (Install / Compile / Run / Debug) (5)

Write a project name and title also set a project folder



## Code Blocks (Install / Compile / Run / Debug) (6)

Select a compiler for this project we selected `gcc` but you can select C compilers from the list. Set Debug and Release executable output folders.





## Code Blocks (Install / Compile / Run / Debug) (7)

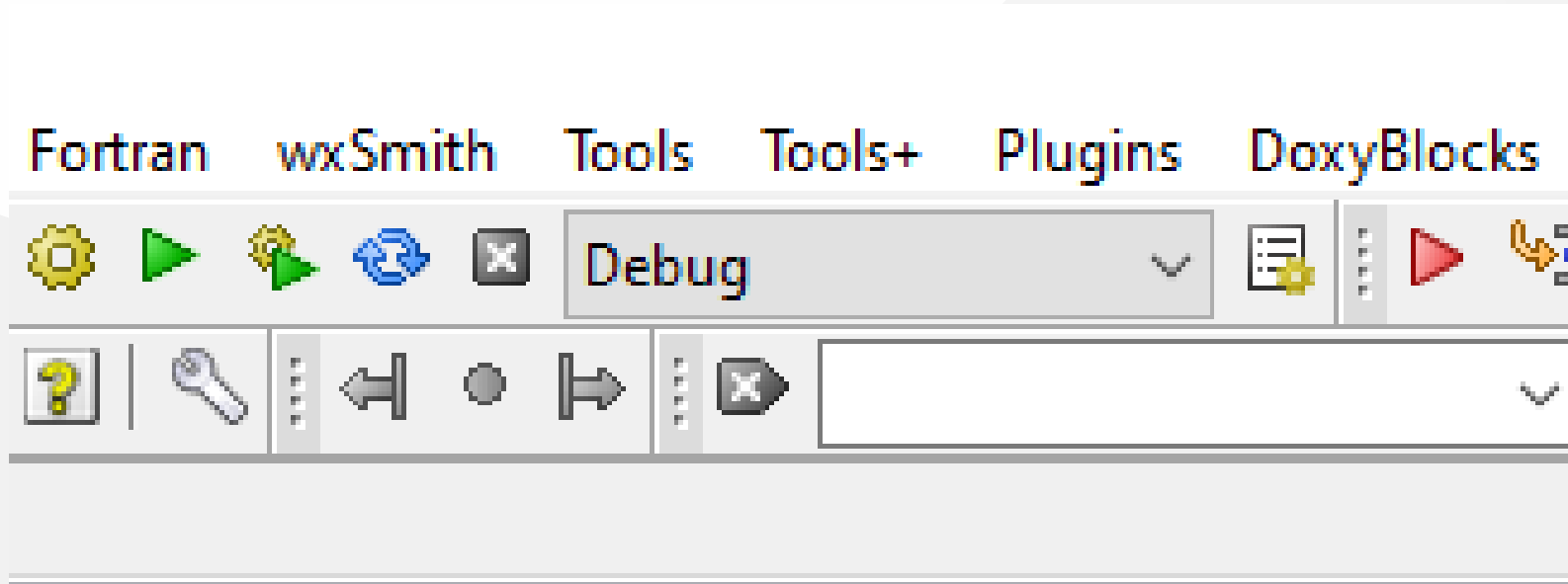
- After this wizard, you will have the following code

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

int main()
{
    printf("Hello world!\n");
    return 0;
}
```

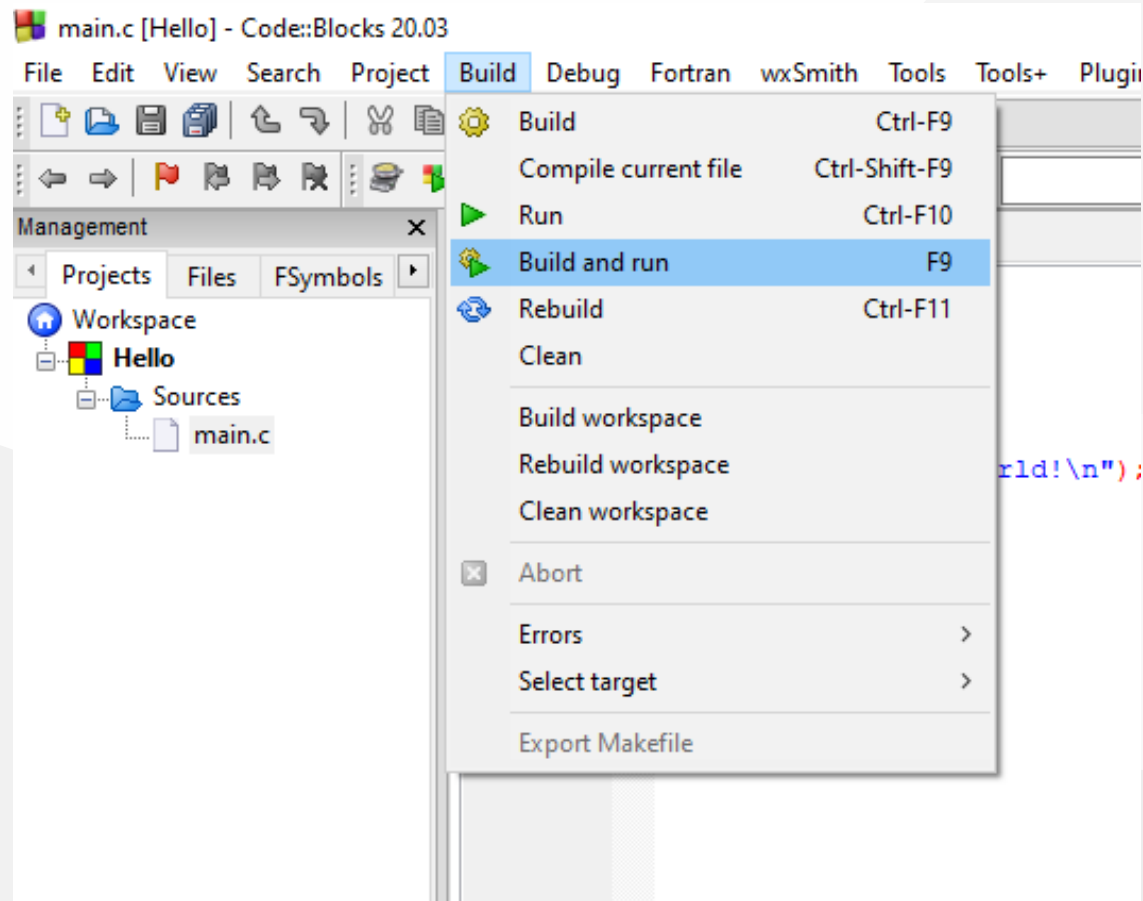
## Code Blocks (Install / Compile / Run / Debug) (8)

Select **Debug** Build from the menu



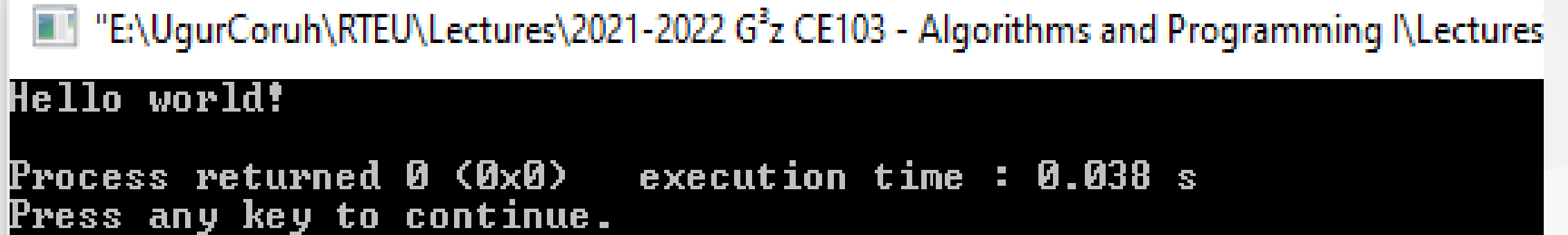
## Code Blocks (Install / Compile / Run / Debug) (9)

Run with Build and Run **F9**



## Code Blocks (Install / Compile / Run / Debug) (10)

- You should see the following output



The screenshot shows a code editor window with the following text:

```
"E:\UgurCoruh\RTEU\Lectures\2021-2022 G3z CE103 - Algorithms and Programming I\Lectures  
Hello world?  
Process returned 0 (0x0)   execution time : 0.038 s  
Press any key to continue.
```

## Code Blocks (Install / Compile / Run / Debug) (11)

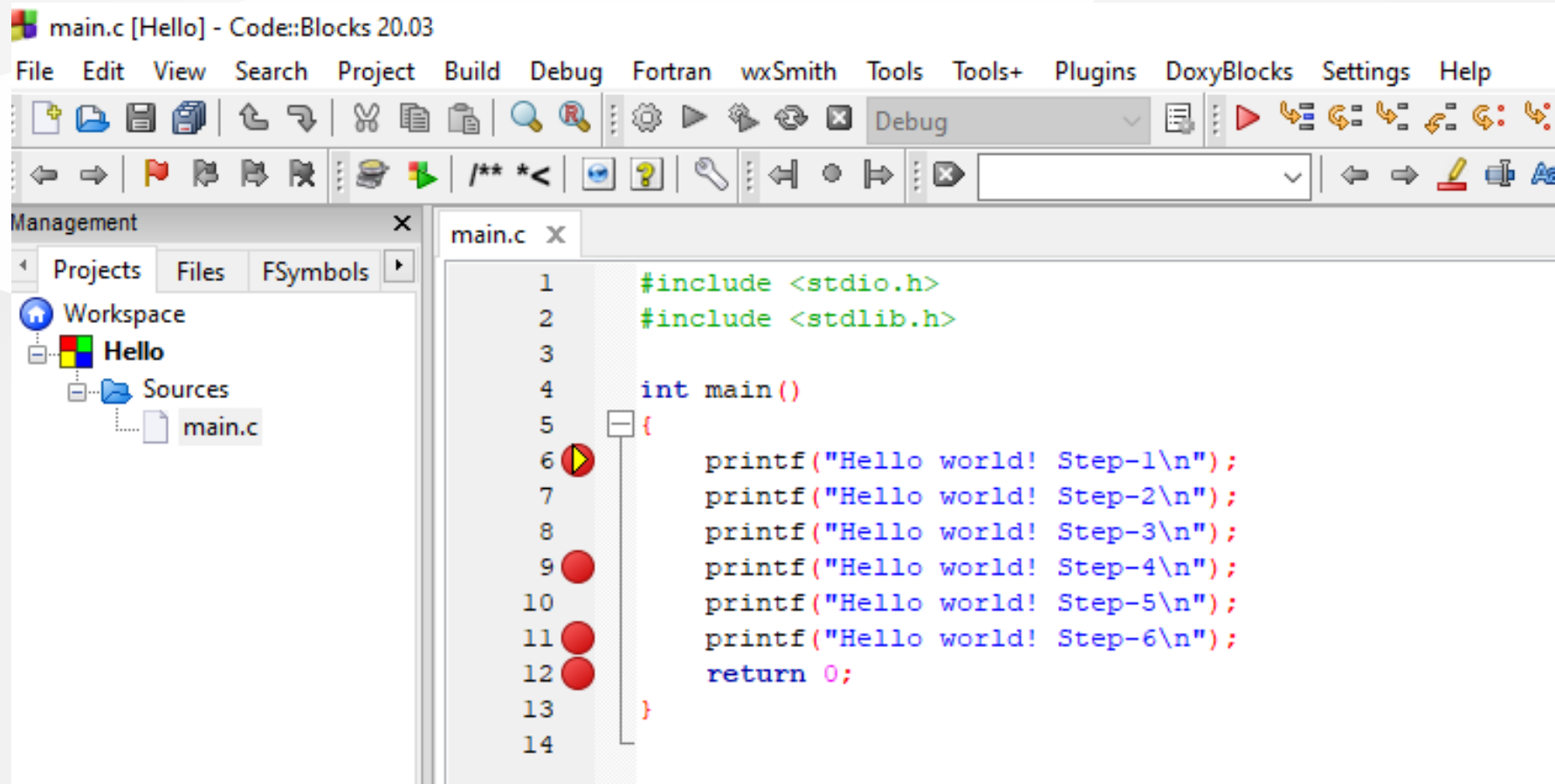
- Add the following lines to your source code for debugging

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

int main()
{
    printf("Hello world! Step-1\n");
    printf("Hello world! Step-2\n");
    printf("Hello world! Step-3\n");
    printf("Hello world! Step-4\n");
    printf("Hello world! Step-5\n");
    printf("Hello world! Step-6\n");
    return 0;
}
```

## Code Blocks (Install / Compile / Run / Debug) (12)

- and add breakpoints with **F5** or mouse click



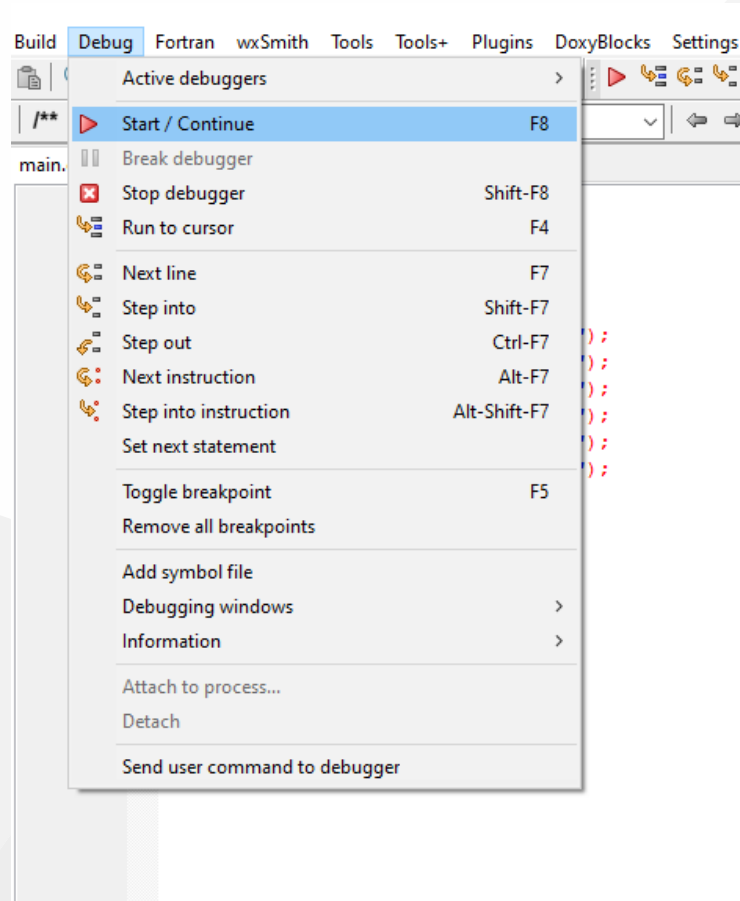
The screenshot shows the Code::Blocks IDE interface. The main window displays a C program named `main.c` with the following code:

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6  printf("Hello world! Step-1\n");
7  printf("Hello world! Step-2\n");
8  printf("Hello world! Step-3\n");
9  printf("Hello world! Step-4\n");
10 printf("Hello world! Step-5\n");
11 printf("Hello world! Step-6\n");
12 return 0;
13 }
14
```

Breakpoints are indicated by red circles on the left margin of the code editor, positioned at lines 6, 9, 10, 11, and 12. A yellow triangle icon is also visible on line 6. The left sidebar shows the project structure with a workspace named "Hello" containing a "Sources" folder with the file `main.c`.

## Code Blocks (Install / Compile / Run / Debug) (13)

- select **Debug->Start/Continue** to start debugger



## Code Blocks (Install / Compile / Run / Debug) (14)

- If you see the following error this is related to long or turkish characters including the path. Just move the project to a shorter path and try again

Setting breakpoints

Debugger name and version: GNU gdb (GDB) 8.1

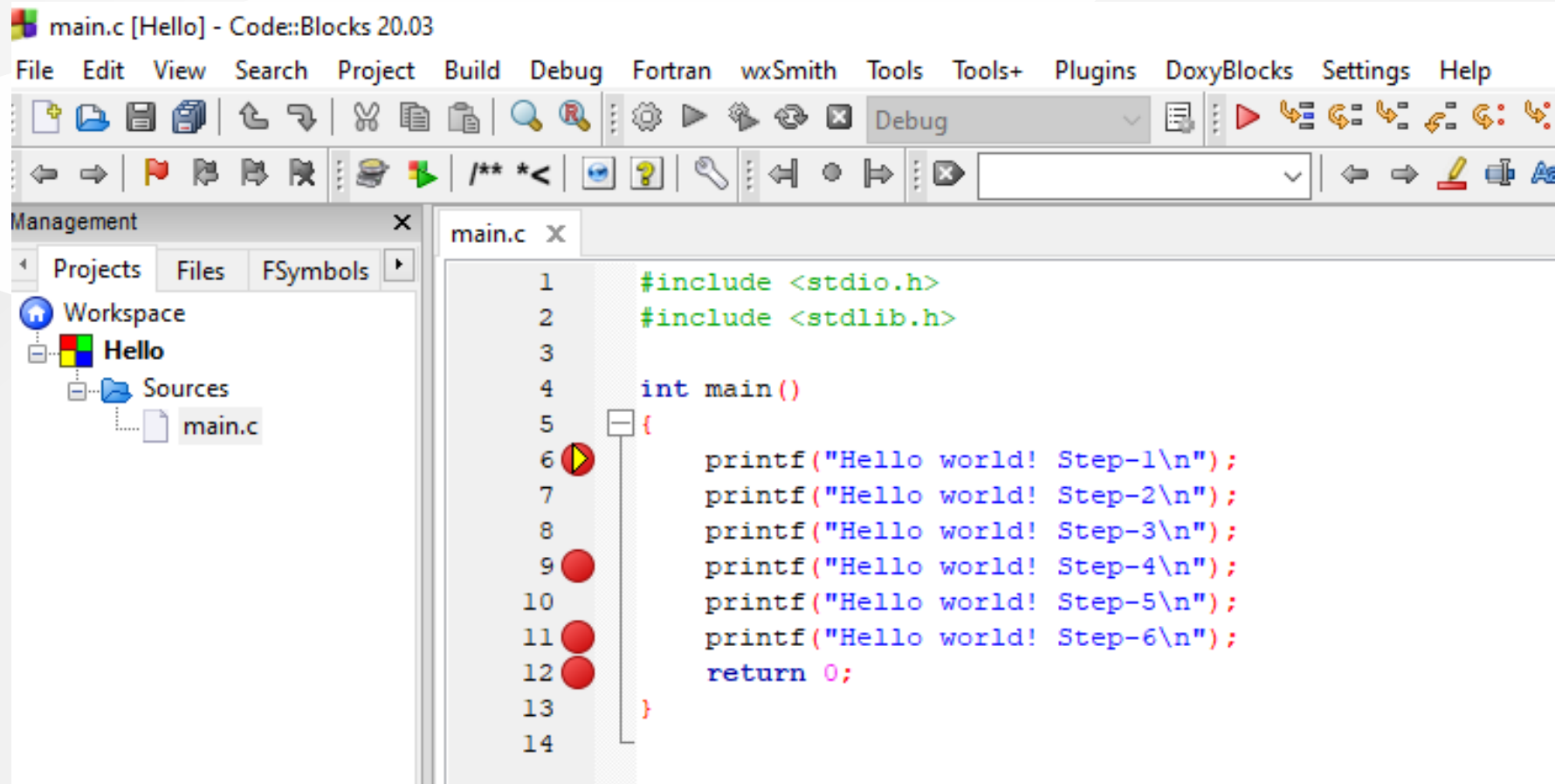
Starting the debuggee failed: No executable specified, use `target exec`.

Debugger finished with status 0



## Code Blocks (Install / Compile / Run / Debug) (15)

You will see the following yellow pointer for the debugger



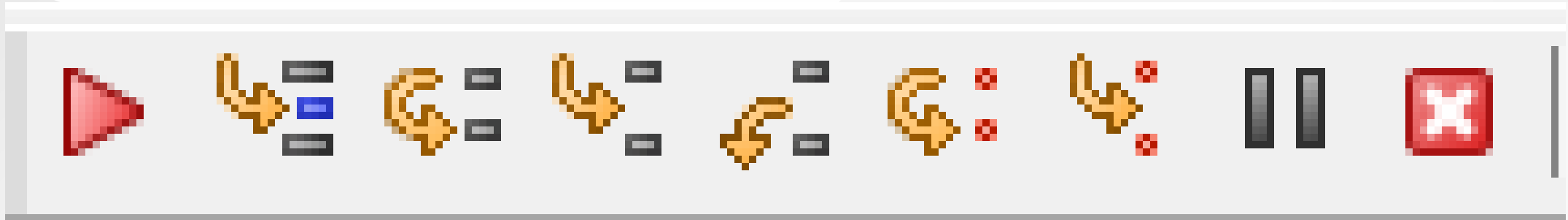
The screenshot shows the Code::Blocks IDE interface. The title bar reads "main.c [Hello] - Code::Blocks 20.03". The menu bar includes File, Edit, View, Search, Project, Build, Debug, Fortran, wxSmith, Tools, Tools+, Plugins, DoxyBlocks, Settings, and Help. The toolbar contains various icons for file operations, compilation, and debugging. The left sidebar shows a project tree with "Workspace", "Hello", and "Sources" containing "main.c". The main editor window displays the following C code:

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6  printf("Hello world! Step-1\n");
7  printf("Hello world! Step-2\n");
8  printf("Hello world! Step-3\n");
9  printf("Hello world! Step-4\n");
10 printf("Hello world! Step-5\n");
11 printf("Hello world! Step-6\n");
12 return 0;
13 }
14
```

A yellow pointer icon is positioned on the left margin of line 6. Red circular markers are visible on lines 9, 10, 11, and 12, indicating breakpoints.

## Code Blocks (Install / Compile / Run / Debug) (16)

You can use the following menu or shortcuts for step-by-step debugging.



## GCC/G++ Compiler (MinGW) / Clang-cl (LLVM) (1)

Download and install `MinGW` or `LLVM` compiler (if you downloaded then skip this step)

- MinGW installer (clang)
  - [Download MinGW-w64 - for 32 and 64-bit Windows from SourceForge.net](#)
- If you have a problem try `Github` builds
  - <https://github.com/nixman/mingw-builds-binaries/releases>
  - [https://github.com/nixman/mingw-builds-binaries/releases/download/12.2.0-rt\\_v10-rev0/x86\\_64-12.2.0-release-win32-seh-rt\\_v10-rev0.7z](https://github.com/nixman/mingw-builds-binaries/releases/download/12.2.0-rt_v10-rev0/x86_64-12.2.0-release-win32-seh-rt_v10-rev0.7z)
- LLVM installer (gcc/g++)
- [Download LLVM releases](#)
  - Also use the following notes
    - <https://llvm.org/devmtg/2014-04/PDFs/Talks/clang-cl.pdf>

## GCC/G++ Compiler (MinGW) / Clang-cl (LLVM) (2)

Open a console with "cmd" and test the following commands if commands are not recognized then set the system environment variable to add gcc and g++ executable paths to the path variable (add to both system and user path variable)

```
gcc --version
```

```
g++ --version
```

```
C:\Users\ugur.coruh>gcc --version
gcc (x86_64-win32-seh-rev0, Built by MinGW-W64 project) 8.1.0
Copyright (C) 2018 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

```
clang --version
```

## GCC/G++ Compiler (MinGW) / Clang-cl (LLVM) (3)

- for `gcc.exe` , `g++.exe` and `gdb.exe`

```
C:\Program Files\mingw-w64\x86_64-8.1.0-win32-seh-rt_v6-rev0\mingw64\bin
```

- for `clang.exe` , `lldb.exe`

```
C:\Program Files\LLVM\bin
```

This folder path changes according to your setup

## **VSCode (Install / Compile / Run / Debug) (1)**

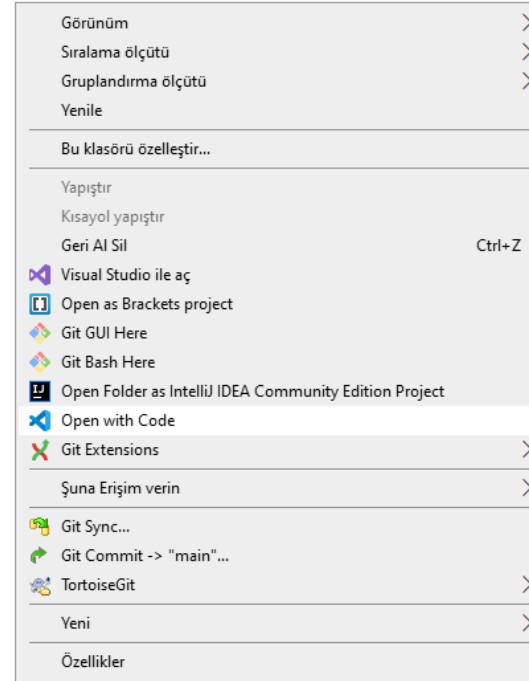
Download Visual Studio Code from the following link

[Download Visual Studio Code - Mac, Linux, Windows](#)

## VSCode (Install / Compile / Run / Debug) (2)

In this sample, you will find MinGW and LLVM compiler combinations for C and C++

Create a folder and enter this folder then open this folder with vscode by right click



## VSCode (Install / Compile / Run / Debug) (3)

or enter the folder via console

```
Güz C E:\UgurCoruh\RTEU\Lectures\2021-2022 Güz CE103 - Algorithms and Programming I\Lectures\ce103-algorithms-and-programming-  
I\Week-2\vscode-hello-world-apps\C-clang>code .  
lo-w  
lo-w  
erson
```

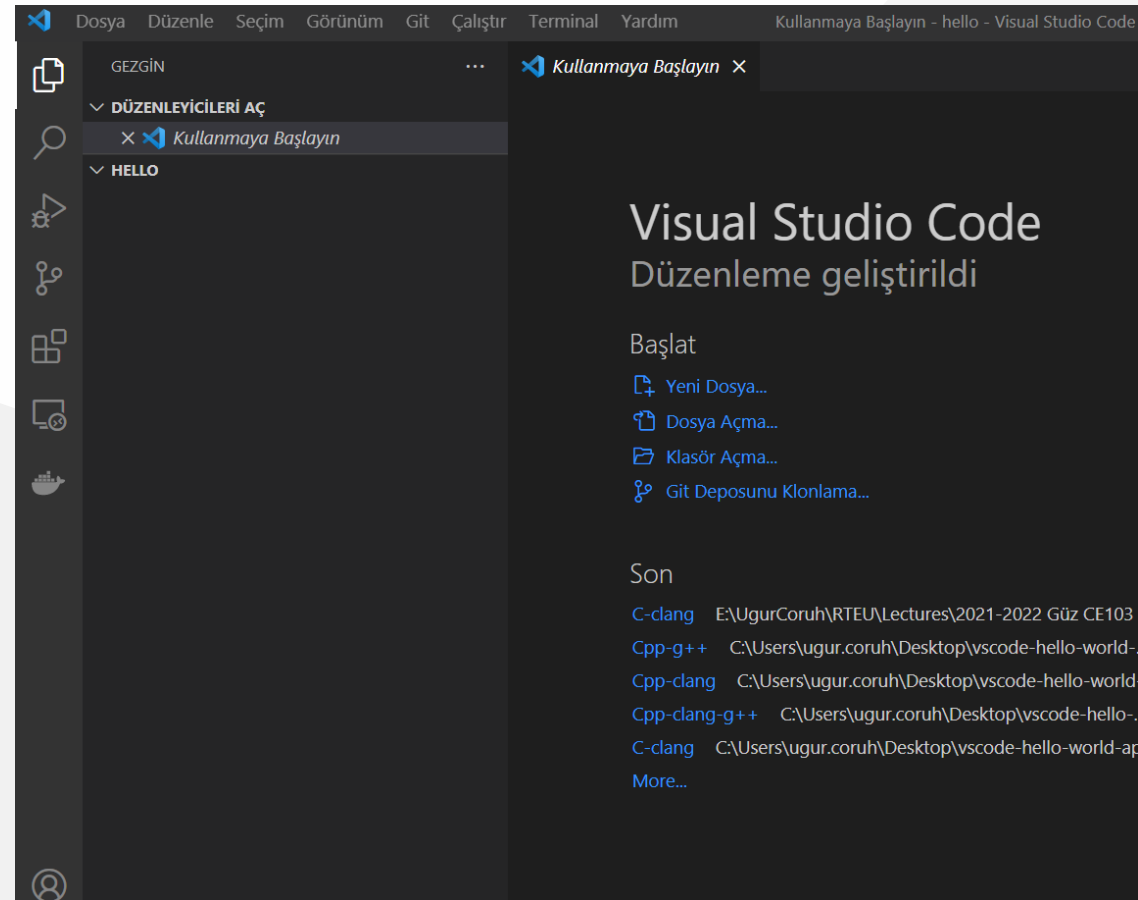
write

```
code .
```

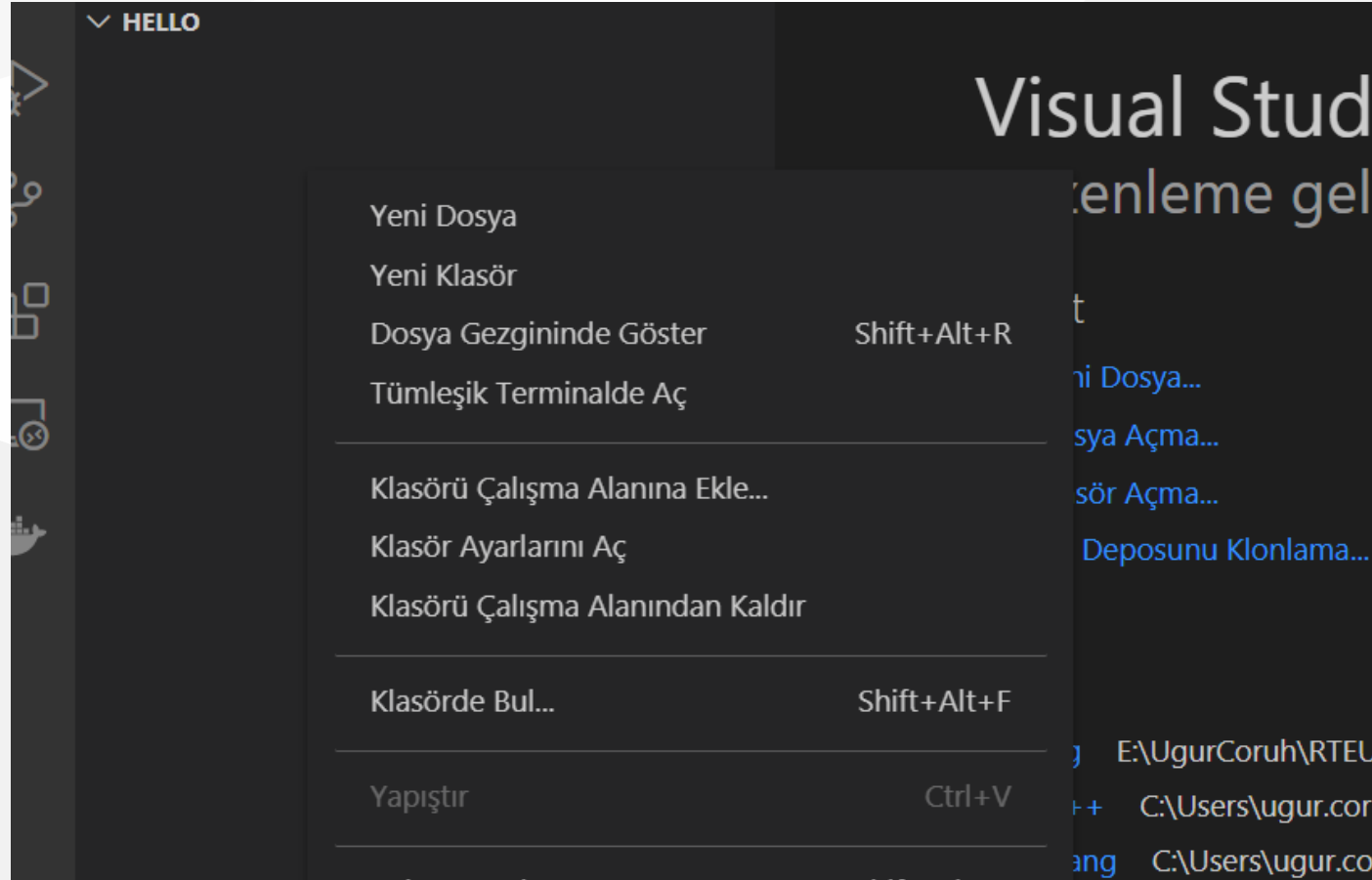


## VSCoDe (Install / Compile / Run / Debug) (4)

- This will open vscode for the current folder, (.) dot present current folder.
- You will see an empty folder in the right window



## VSCoDe (Install / Compile / Run / Debug) (5)



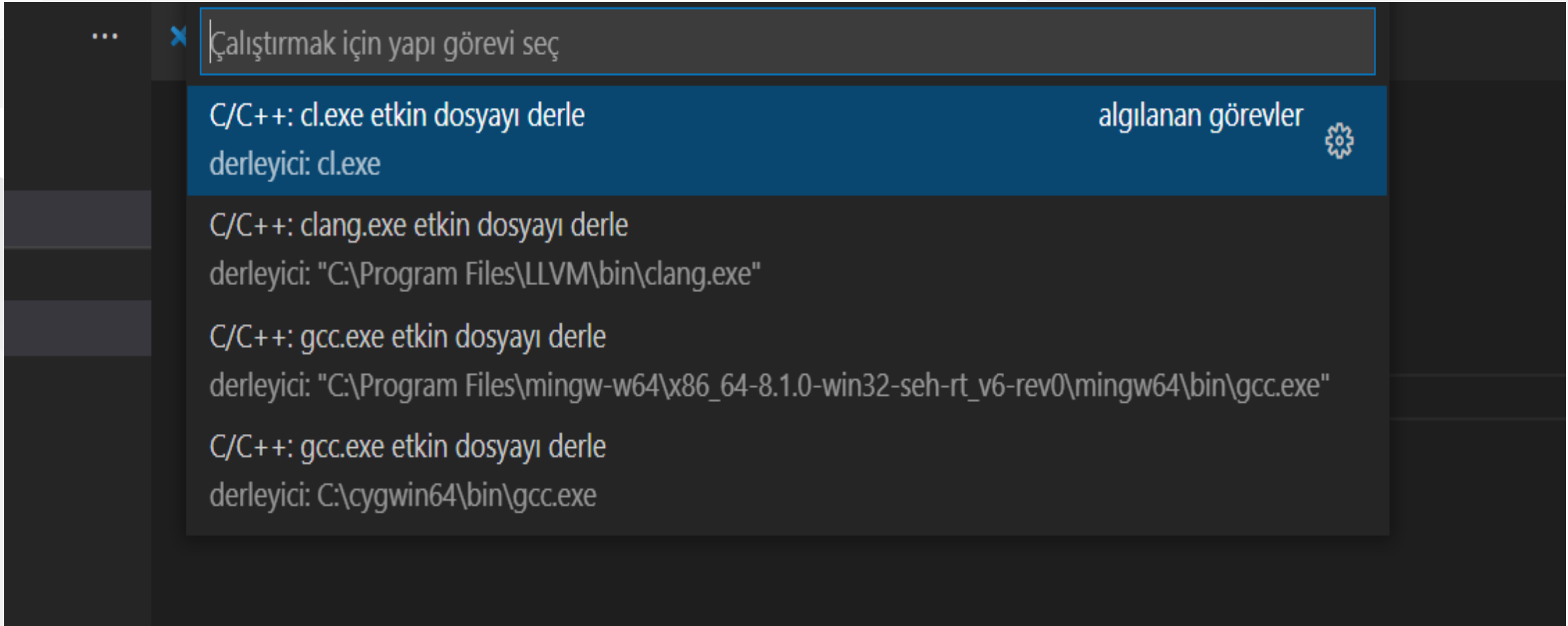
## VSCoDe (Install / Compile / Run / Debug) (6)

- Create a `hello.c` file and write the following content

```
#include <stdio.h>
int main() {
    // printf() displays the string inside quotation
    printf("Hello, World!");
    return 0;
}
```

## VSCoDe (Install / Compile / Run / Debug) (7)

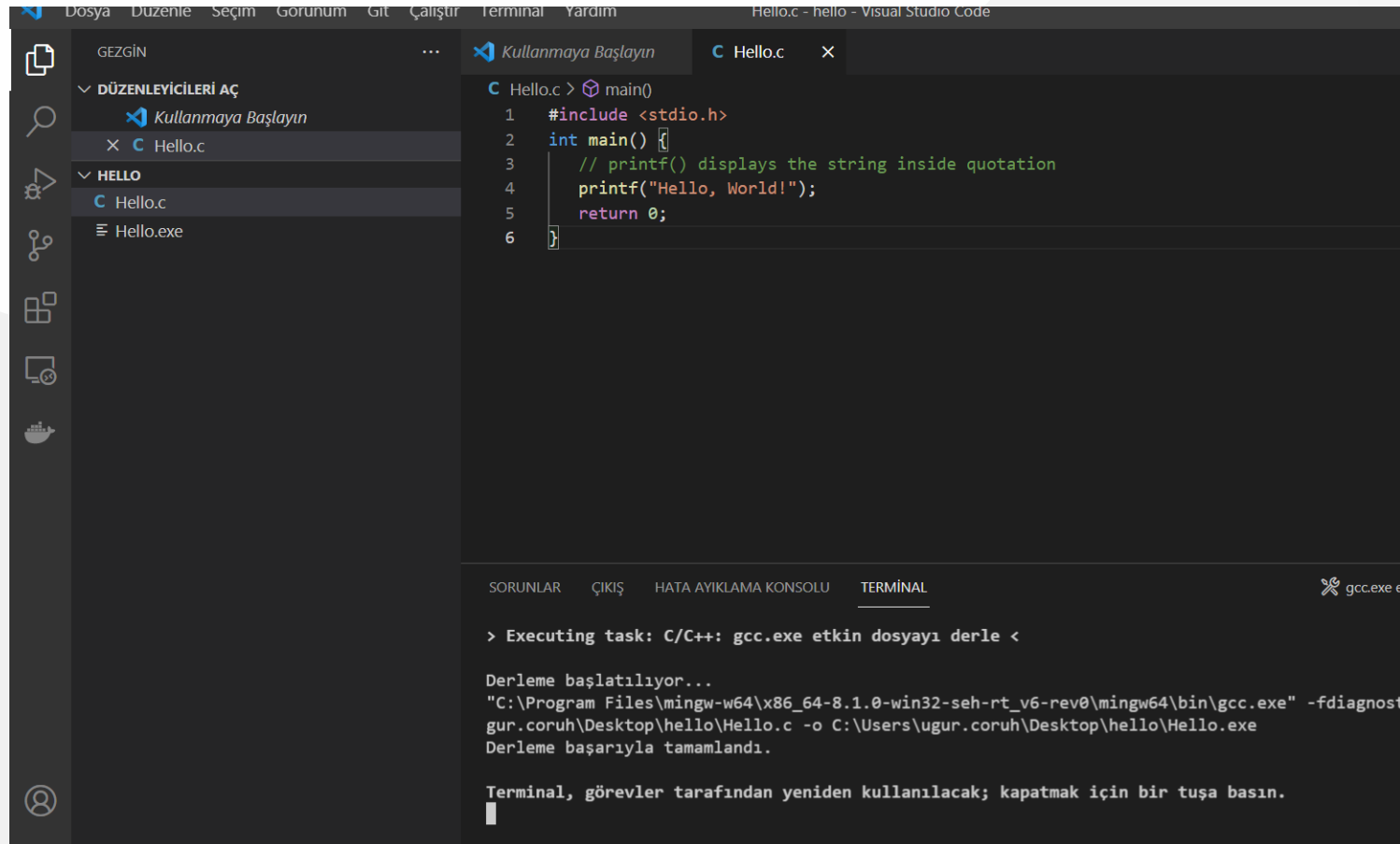
use `CTRL+SHIFT+B` (you should be on the source code section) to build a file



## VSCoDe (Install / Compile / Run / Debug) (8)

Select `GCC` or `CLANG` for this sample we can use `GCC`

You will see the output generated `Hello.exe`



```

Dosya  Düzenle  Seçim  Görünüm  Git  Çalıştır  Terminal  Yardım  Hello.c - hello - Visual Studio Code
GEZGİN
DÜZENLEYİCİLERİ AÇ
  Kullanmaya Başlayın
  C Hello.c
HELLO
  C Hello.c
  Hello.exe
C Hello.c > main()
1 #include <stdio.h>
2 int main()
3 // printf() displays the string inside quotation
4 printf("Hello, World!");
5 return 0;
6 }

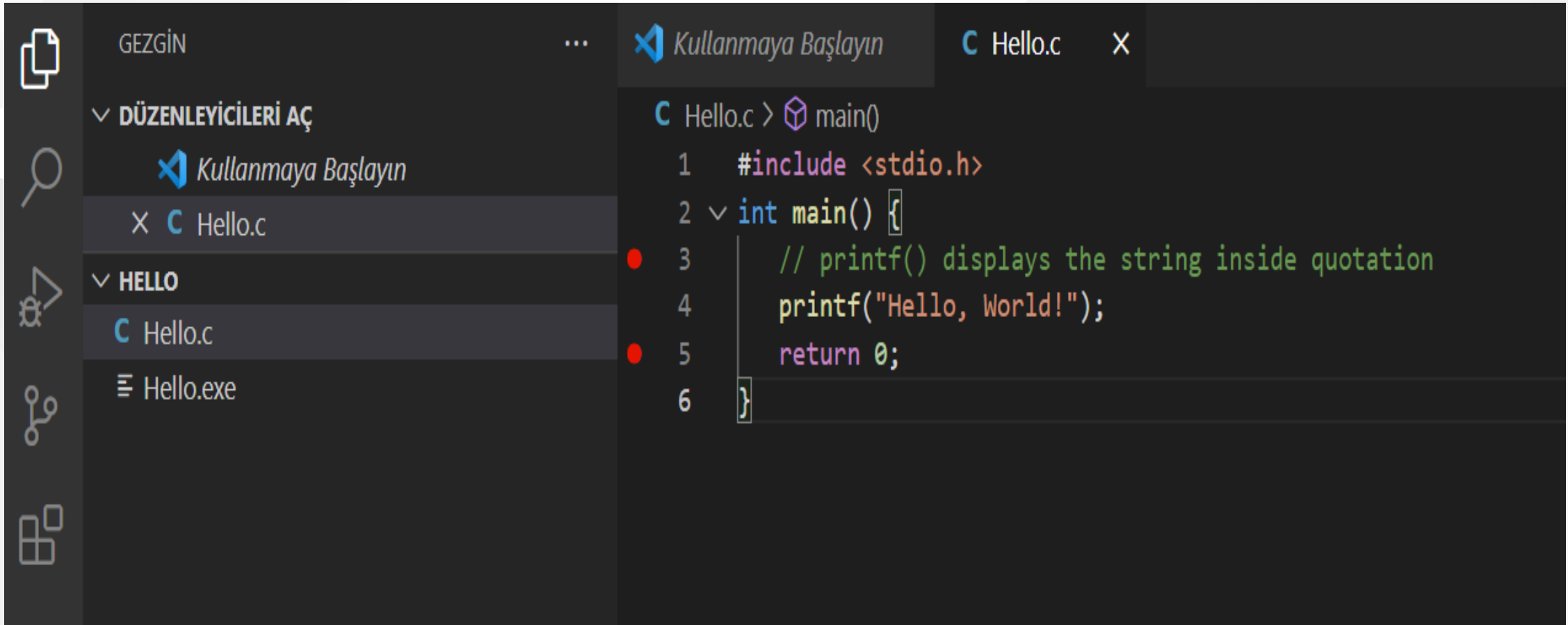
SORUNLAR  ÇIKIŞ  HATA AYIKLAMA KONSOLU  TERMINAL  gcc.exe e
> Executing task: C/C++: gcc.exe etkin dosyayı derle <

Derleme başlatılıyor...
"C:\Program Files\mingw-w64\x86_64-8.1.0-win32-seh-rt_v6-rev0\mingw64\bin\gcc.exe" -fdiagnost
gur.coruh\Desktop\hello\Hello.c -o C:\Users\ugur.coruh\Desktop\hello\Hello.exe
Derleme başarıyla tamamlandı.

Terminal, görevler tarafından yeniden kullanılacak; kapatmak için bir tuşa basın.
```

## VSCoDe (Install / Compile / Run / Debug) (9)

for debugging just put a breakpoint and build again

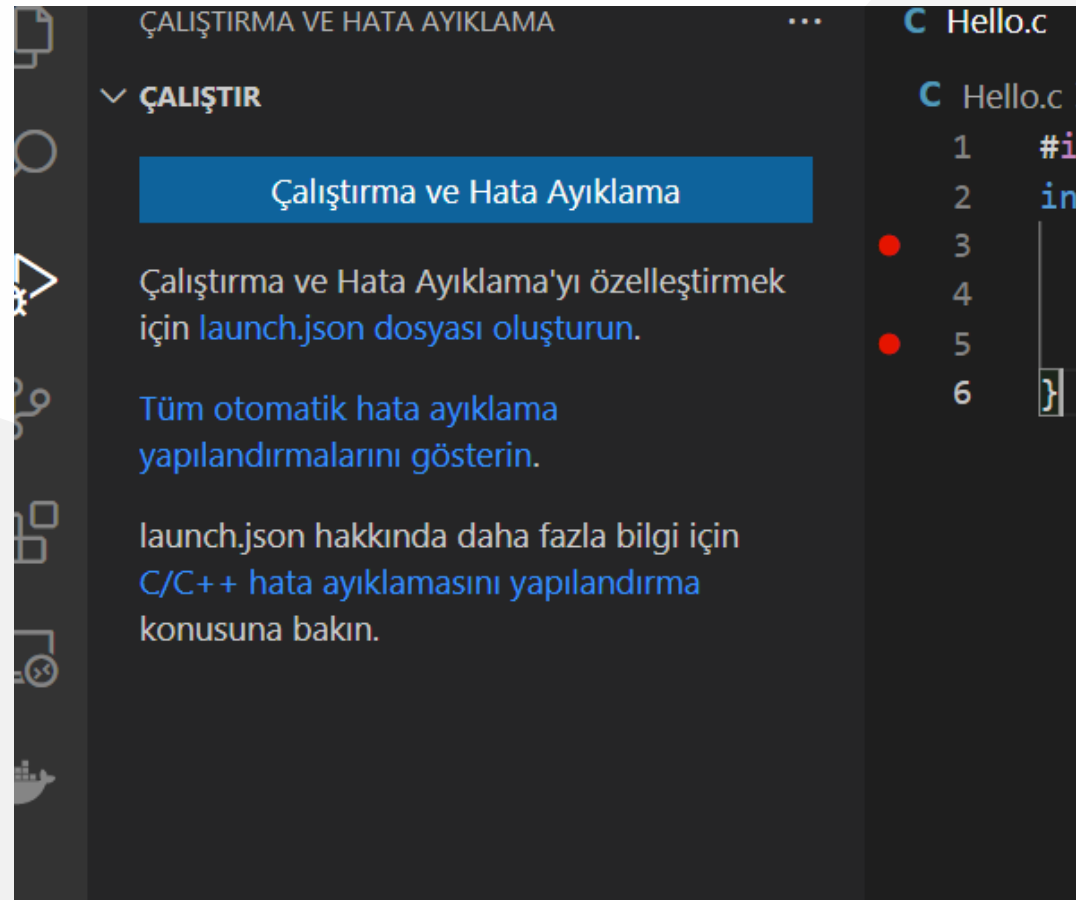


The screenshot shows the Visual Studio Code interface. The left sidebar contains the Explorer, Search, Run and Debug, and Extensions views. The Explorer view shows a project named 'GEZGIN' with a folder 'DÜZENLEYİCİLERİ AÇ' containing 'Hello.c' and a folder 'HELLO' containing 'Hello.c' and 'Hello.exe'. The main editor area shows the code for 'Hello.c' with a breakpoint set on line 3. The code is as follows:

```
C Hello.c > main()
1  #include <stdio.h>
2  int main() {
3      // printf() displays the string inside quotation
4      printf("Hello, World!");
5      return 0;
6  }
```

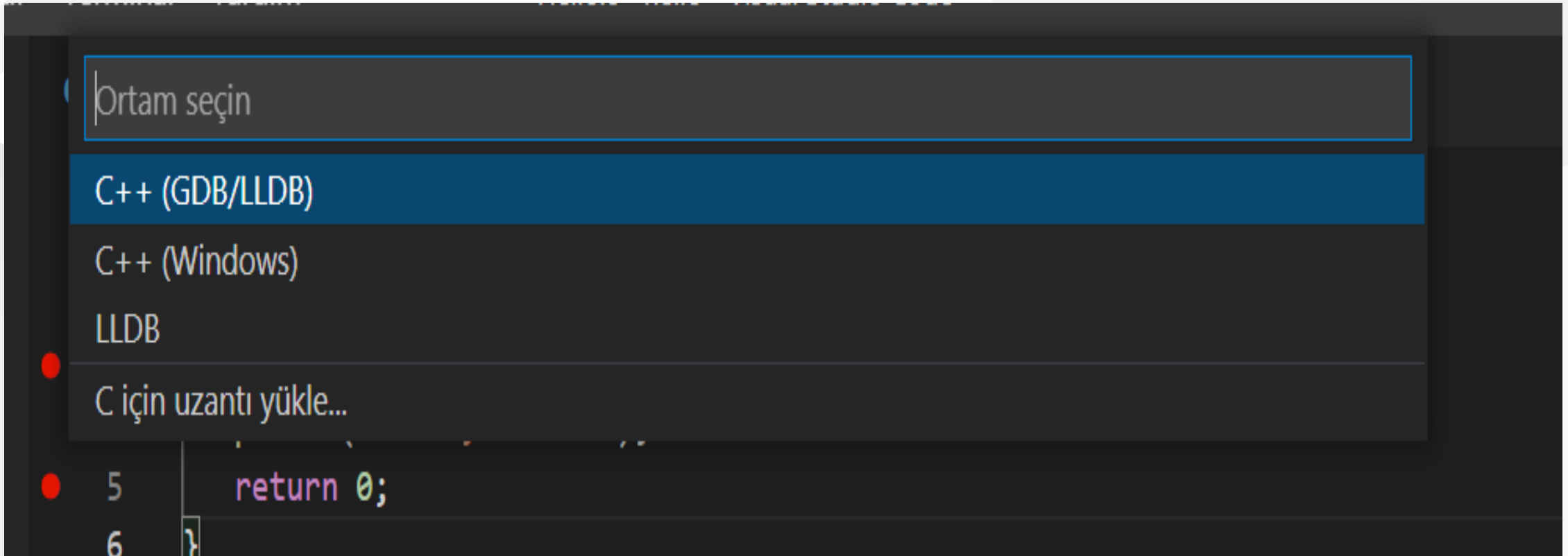
## VSCode (Install / Compile / Run / Debug) (10)

- after building for debug press `CTRL+SHIFT+D` (you should be in the source code section) and in the right window select create `launch.json`



## VSCoDe (Install / Compile / Run / Debug) (11)

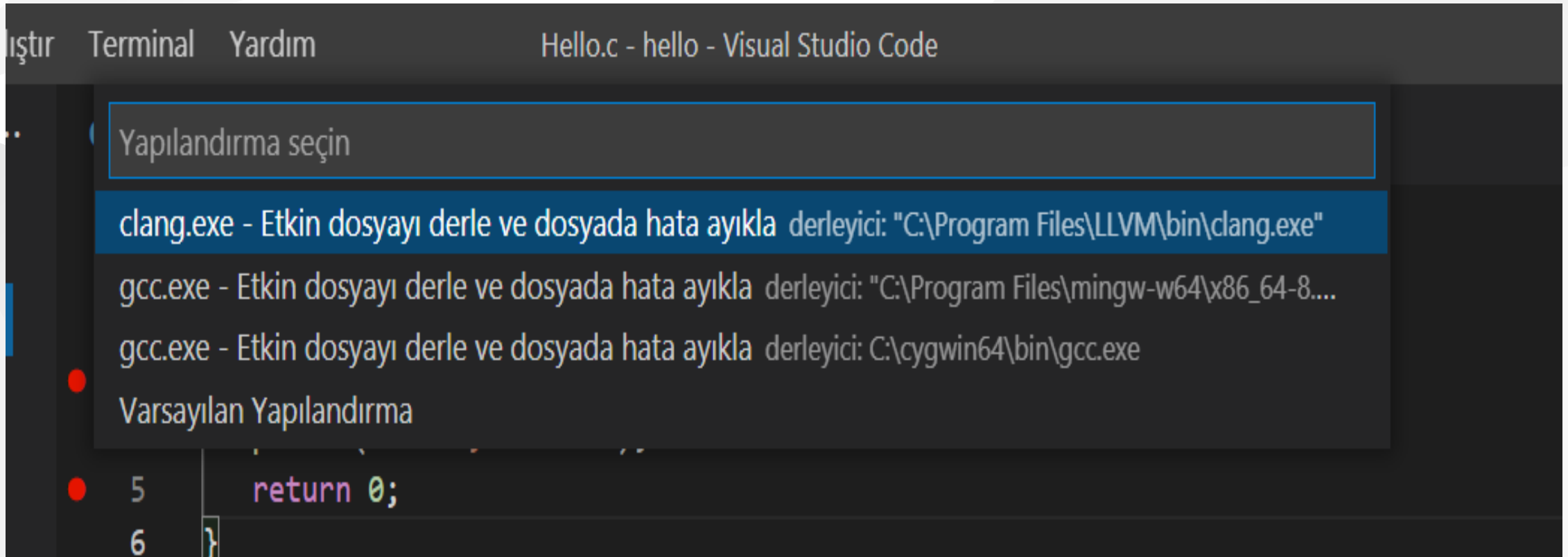
- from opening, window select `C++ GDB/LLDB`





## VSCoDe (Install / Compile / Run / Debug) (12)

- from the next opening, menu select `mingw-w64 gcc.exe`



The screenshot shows the Visual Studio Code interface with the 'Hello.c - hello - Visual Studio Code' window open. The 'Terminal' menu is open, and the 'Yapılandırma seçin' (Select Compiler) option is highlighted. The menu lists several compiler options:

- clang.exe - Etkin dosyayı derle ve dosyada hata ayıkla derleyici: "C:\Program Files\LLVM\bin\clang.exe"
- gcc.exe - Etkin dosyayı derle ve dosyada hata ayıkla derleyici: "C:\Program Files\mingw-w64\x86\_64-8...."
- gcc.exe - Etkin dosyayı derle ve dosyada hata ayıkla derleyici: C:\cygwin64\bin\gcc.exe
- Varsayılan Yapılandırma

The code editor shows the following code:

```
5     return 0;  
6 }
```

## VSCoDe (Install / Compile / Run / Debug) (13)

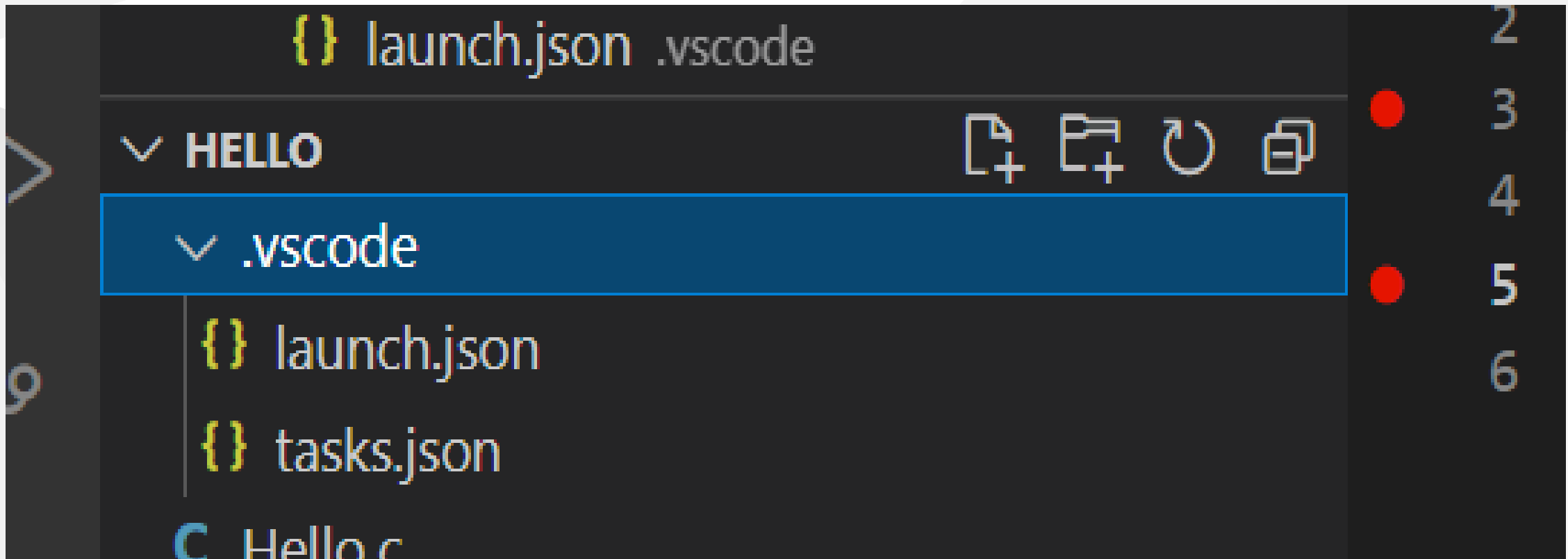
this will run the debugger and you will see debug points activated

```
Dosya Düzenle Seçim Görünüm Git Çalıştır Terminal Yardım Hello.c - hello - Visual Studio Code
ÇALIŞTIRMA V... gcc.exe - Etkir
DEĞİŞKENLER
  Locals
  Registers
Hello.c > main()
1 #include <stdio.h>
2 int main() {
3 // printf() displays the string inside quotation
4 printf("Hello, World!");
5 return 0;
6 }
```

## VSCode (Install / Compile / Run / Debug) (14)

then you can step-by-step debug your code.

the following `task.json` and `launch.json` automatically generated with your selections



# VSCode (Install / Compile / Run / Debug) (15)

## launch.json

```
{
  // Olası öznitelikler hakkında bilgi edinmek için IntelliSense kullanın.
  // Mevcut özniteliklerin açıklamalarını görüntülemek için üzerine gelin.
  // Daha fazla bilgi için şu adresi ziyaret edin: https://go.microsoft.com/fwlink/?linkid=830387
  "version": "0.2.0",
  "configurations": [
    {
      "name": "gcc.exe - Etkin dosyayı derle ve dosyada hata ayıkla",
      "type": "cppdbg",
      "request": "launch",
      "program": "${fileDirname}\\${fileBasenameNoExtension}.exe",
      "args": [],
      "stopAtEntry": false,
      "cwd": "${fileDirname}",
      "environment": [],
      "externalConsole": false,
      "MIMode": "gdb",
      "miDebuggerPath": "C:\\Program Files\\mingw-w64\\x86_64-8.1.0-win32-seh-rt_v6-rev0\\mingw64\\bin\\gdb.exe",
      "setupCommands": [
        {
          "description": "gdb için düzgün yazdırmayı etkinleştir",
          "text": "-enable-pretty-printing",
          "ignoreFailures": true
        }
      ],
      "preLaunchTask": "C/C++: gcc.exe etkin dosyayı derle"
    }
  ]
}
```

## VSCode (Install / Compile / Run / Debug) (16)

## task.json

```
{
  "tasks": [
    {
      "type": "cppbuild",
      "label": "C/C++: gcc.exe etkin dosyayı derle",
      "command": "C:\\Program Files\\mingw-w64\\x86_64-8.1.0-win32-seh-rt_v6-rev0\\mingw64\\bin\\gcc.exe",
      "args": [
        "-fdiagnostics-color=always",
        "-g",
        "${file}",
        "-o",
        "${fileDirname}\\${fileBasenameNoExtension}.exe"
      ],
      "options": {
        "cwd": "${fileDirname}"
      },
      "problemMatcher": ["$gcc"],
      "group": {
        "kind": "build",
        "isDefault": true
      },
      "detail": "Hata Ayıklayıcısı tarafından oluşturulan görev."
    }
  ],
  "version": "2.0.0"
}
```

## VSCode (Install / Compile / Run / Debug) (17)

- You can do the same thing for other compilers and C++ source codes. LLVM does not support debugging on vscode now.

for C++ VsCode you can check the following links

- for Windows
  - <https://code.visualstudio.com/docs/cpp/config-mingw>
- for Linux
  - <https://code.visualstudio.com/docs/cpp/config-linux>
- for WSL
  - <https://code.visualstudio.com/docs/cpp/config-wsl>

## VSCoDe (Install / Compile / Run / Debug) (18)

in the launch file if you start debugging with **F5**

(you can select debugger with **CTRL+SHIFT+P** and then write Debug and Selecting Configure Debugger Option)

## VSCoDe (Install / Compile / Run / Debug) (19)

- the following line will be your debugging application path
- if you start debugging with `F5` in `Hello.c` file this will set `<Hello.c base path>/Hello.exe`



## VSCode (Install / Compile / Run / Debug) (20)

You should set this correct for both `LLVM` and `GCC` configuration in `launch.json`

```
"program": "${fileDirname}\\${fileBasenameNoExtension}.exe",
```

Also you should set your installed debugger paths

for GCC

```
"miDebuggerPath": "C:\\Program Files\\mingw-w64\\x86_64-8.1.0-win32-seh-rt_v6-rev0\\mingw64\\bin\\gdb.exe",
```

for LLVM

```
"miDebuggerPath": "C:\\Program Files\\LLVM\\bin\\lldb.exe",
```

for more details please check the sample source codes.

## Visual Studio Code Extension List (1)

### My Extension List

- Listing Installed Extensions

```
PS C:\Users\ugur.coruh\Desktop> code --list-extensions | % { "code --install-extension $_" }
```

Following topic can help you

[How can you export the Visual Studio Code extension list? - Stack Overflow](#)

## Visual Studio Code Extension List (2)

```
code --install-extension 2gua.rainbow-brackets
code --install-extension aaron-bond.better-comments
code --install-extension abusaidm.html-snippets
code --install-extension ACharLuk.easy-cpp-projects
code --install-extension chris-noring.node-snippets
code --install-extension cschlosser.doxdocgen
code --install-extension csholmq.excel-to-markdown-table
code --install-extension DaChuiOpenSource.FreeMind
code --install-extension dannysteenman.cloudformation-yaml-snippets
code --install-extension Dart-Code.dart-code
code --install-extension Dart-Code.flutter
code --install-extension digized.umple
code --install-extension DotJoshJohnson.xml
code --install-extension DougFinke.vscode-pandoc
code --install-extension dzhavat.bracket-pair-toggler
code --install-extension esbenp.prettier-vscode
code --install-extension formulahendry.dotnet
code --install-extension franneck94.c-cpp-runner
code --install-extension gcc.
```

## Visual Studio Code Extension List (3)

```
vscode-plugin-billionbottle
code --install-extension geeklearningio.graphviz-markdown-preview
code --install-extension geyao.html-snippets
code --install-extension GitHub.copilot-nightly
code --install-extension GrapeCity.gc-excelviewer
code --install-extension Ionide.Ionide-fsharp
code --install-extension ionut-botizanu.vscode-cypher-ql
code --install-extension ipedrazas.kubernetes-snippets
code --install-extension JakeWilson.vscode-picture
code --install-extension James-Yu.latex-workshop
code --install-extension JasonMejane.base64viewer
code --install-extension jasonnutter.search-node-modules
code --install-extension jebbs.plantum1
code --install-extension jeff-hykin.better-cpp-syntax
code --install-extension Katacoda.vscode
code --install-extension KenDomino.Antlrvsix-vscode
code --install-extension l7ssha.tag-inserter
code --install-extension lolkush.quickstart
code --install-extension marp-team.marp-vscode
code --install-extension mindaro-dev.file-downloader
code --install-extension mindaro.mindaro
code --install-extension ms-azuretools.vscode-docker
code --install-extension MS-CEINTL.vscode-language-pack-tr
```

## Visual Studio Code Extension List (4)

```
code --install-extension ms-dotnettools.csharp
code --install-extension ms-dotnettools.dotnet-interactive-vscode
code --install-extension ms-dotnettools.vscode-dotnet-pack
code --install-extension ms-dotnettools.vscode-dotnet-runtime
code --install-extension ms-kubernetes-tools.vscode-aks-tools
code --install-extension ms-kubernetes-tools.vscode-kubernetes-tools
code --install-extension ms-python.python
code --install-extension ms-python.vscode-pylance
code --install-extension ms-toolsai.jupyter
code --install-extension ms-toolsai.jupyter-keymap
code --install-extension ms-toolsai.jupyter-renderers
code --install-extension ms-vscode-remote.remote-containers
code --install-extension ms-vscode-remote.remote-ssh
code --install-extension ms-vscode-remote.remote-ssh-edit
code --install-extension ms-vscode-remote.remote-wsl
```

## Visual Studio Code Extension List (5)

```
code --install-extension ms-vscode.azure-account
code --install-extension ms-vscode.brackets-keybindings
code --install-extension ms-vscode.brackets-pack
code --install-extension ms-vscode.cmake-tools
code --install-extension ms-vscode.cpptools
code --install-extension ms-vscode.cpptools-extension-pack
code --install-extension ms-vscode.cpptools-themes
code --install-extension ms-vscode.live-server
code --install-extension ms-vsiveshare.vsliveshare
code --install-extension oleg-shilo.cs-script
code --install-extension PascalReitermann93.vscode-yaml-sort
```

## Visual Studio Code Extension List (6)

```
code --install-extension Pivotal.vscode-boot-dev-pack
code --install-extension Pivotal.vscode-concourse
code --install-extension Pivotal.vscode-manifest-yaml
code --install-extension Pivotal.vscode-spring-boot
code --install-extension PKief.material-icon-theme
code --install-extension platformio.platformio-ide
code --install-extension pranaygp.vscode-css-peek
code --install-extension redhat.fabric8-analytics
code --install-extension redhat.java
code --install-extension redhat.vscode-commons
code --install-extension redhat.vscode-xml
code --install-extension redhat.vscode-yaml
code --install-extension ritwickdey.LiveServer
code --install-extension sidthesloth.html5-boilerplate
code --install-extension TaodongWu.ejs-snippets
code --install-extension tht13.python
code --install-extension tomoki1207.pdf
code --install-extension twxs.cmake
code --install-extension vadimcn.vscode-lldb
```

## Visual Studio Code Extension List (7)

```
code --install-extension VisualStudioExptTeam.intellicode-api-usage-examples
code --install-extension VisualStudioExptTeam.vscodintellicode
code --install-extension vscjava.vscod-java-debug
code --install-extension vscjava.vscod-java-dependency
code --install-extension vscjava.vscod-java-pack
code --install-extension vscjava.vscod-java-test
code --install-extension vscjava.vscod-maven
code --install-extension vscjava.vscod-spring-boot-dashboard
code --install-extension vscjava.vscod-spring-initializr
code --install-extension walkme.HTML5-extension-pack
code --install-extension webfreak.debug
code --install-extension well-ar.plantum1
code --install-extension wildboar.asn1
code --install-extension Zignd.html-css-class-completion
```



## Visual Studio Community Edition (Install / Compile / Run / Debug) (1)

- Download and install `Visual Studio Community Edition`
- Select All Development Environments from Installer.

### Ücretsiz Geliştirici Yazılımları ve Hizmetleri - Visual Studio



#### Visual Studio Community

Windows | Sürüm 17.2

Windows kullanan .NET ve C++ geliştiricileri için en iyi kapsamlı IDE. Yazılım geliştirmenin her aşamasını yükseltip geliştiren bir dizi güzel araçlar ve özelliklerle tam olarak paketlenmiştir.

Daha fazla bilgi >

Ücretsiz indirin



#### Mac için Visual Studio

Mac | Sürüm 17.

macOS'e özgü .NET geliştiricileri için kapsamlı bir IDE. Web, bulut, mobil ve oyun geliştirme için en üst düzey desteği içerir.

Daha fazla bilgi >

Şunun hakkında daha fazla bilgi edinin:  
[İsansız etkinleştiriliyor](#)

Ücretsiz indirin



#### Visual Studio Code

Windows, macOS, Linux | Sürüm 1.68

Windows, macOS ve Linux üzerinde çalışan tek başına bir kaynak kod düzenleyicisi. JavaScript ve web geliştiricileri için hemen hemen her programlama dilini destekleyecek uzantılara sahip en iyi seçim.

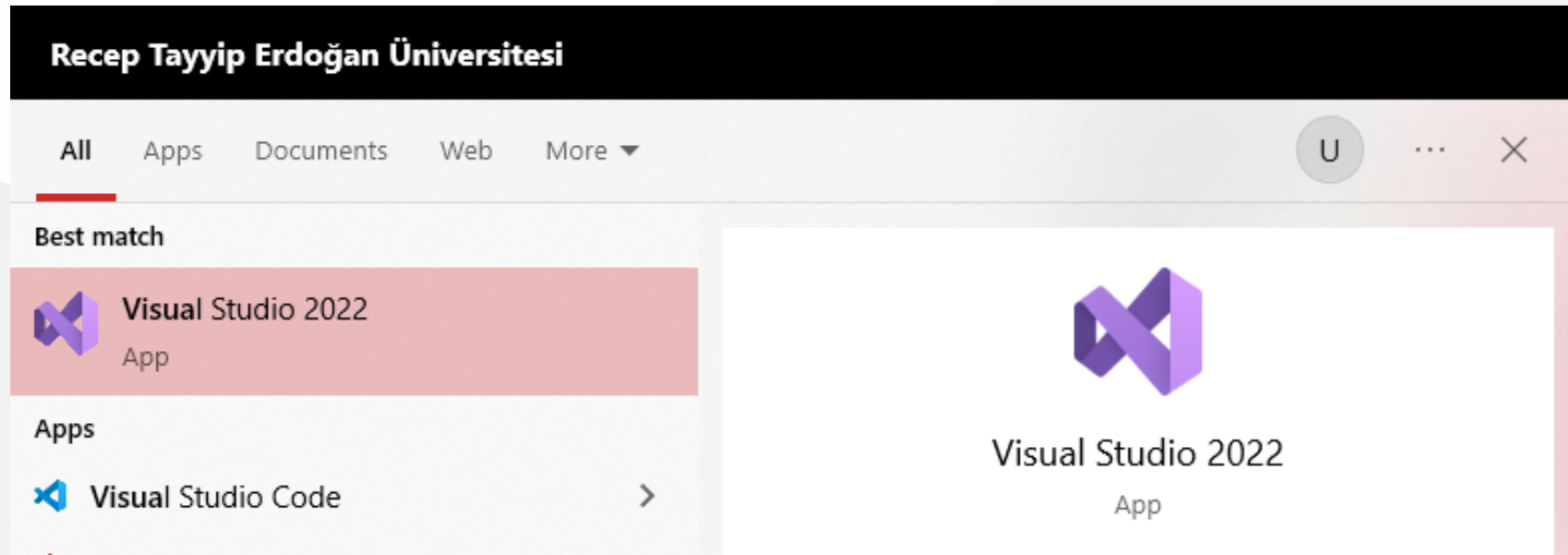
Daha fazla bilgi >

Visual Studio Code'u kullanarak [İsans & gizlilik bildiri](#)

Ücretsiz indirin

## Visual Studio Community Edition (Install / Compile / Run / Debug) (2)

- After installation open `Visual Studio 2022`` from the menu.



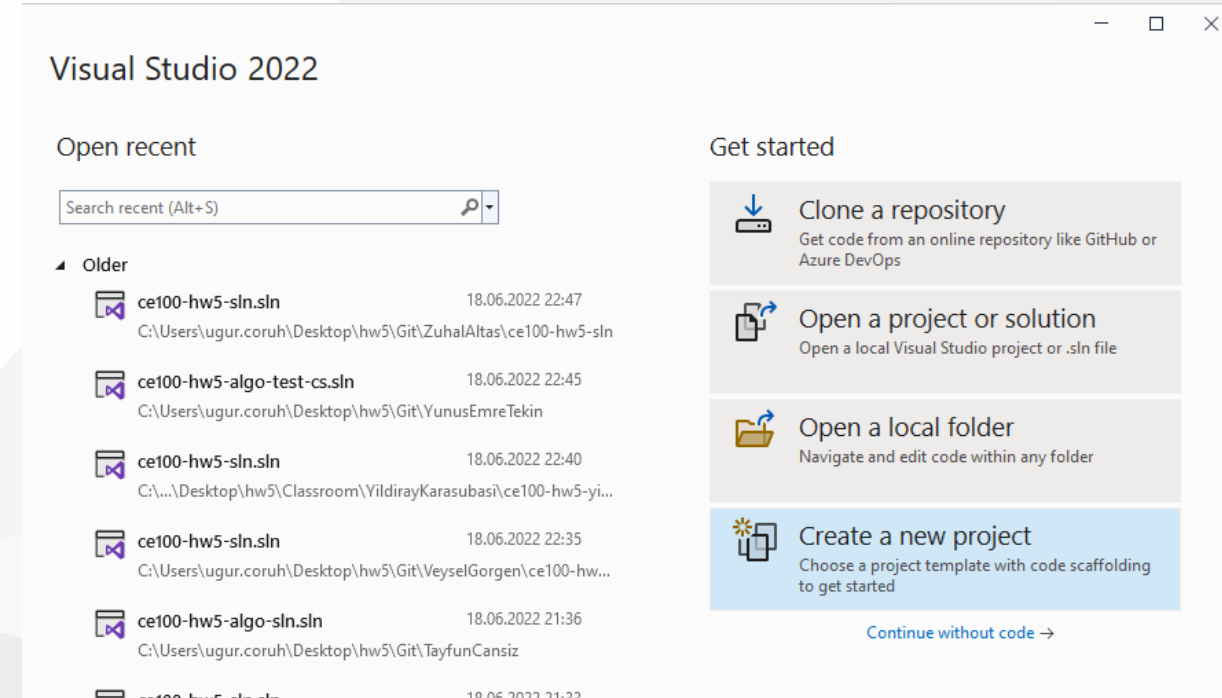
## Visual Studio Community Edition (Install / Compile / Run / Debug) (3)

- The application will start...



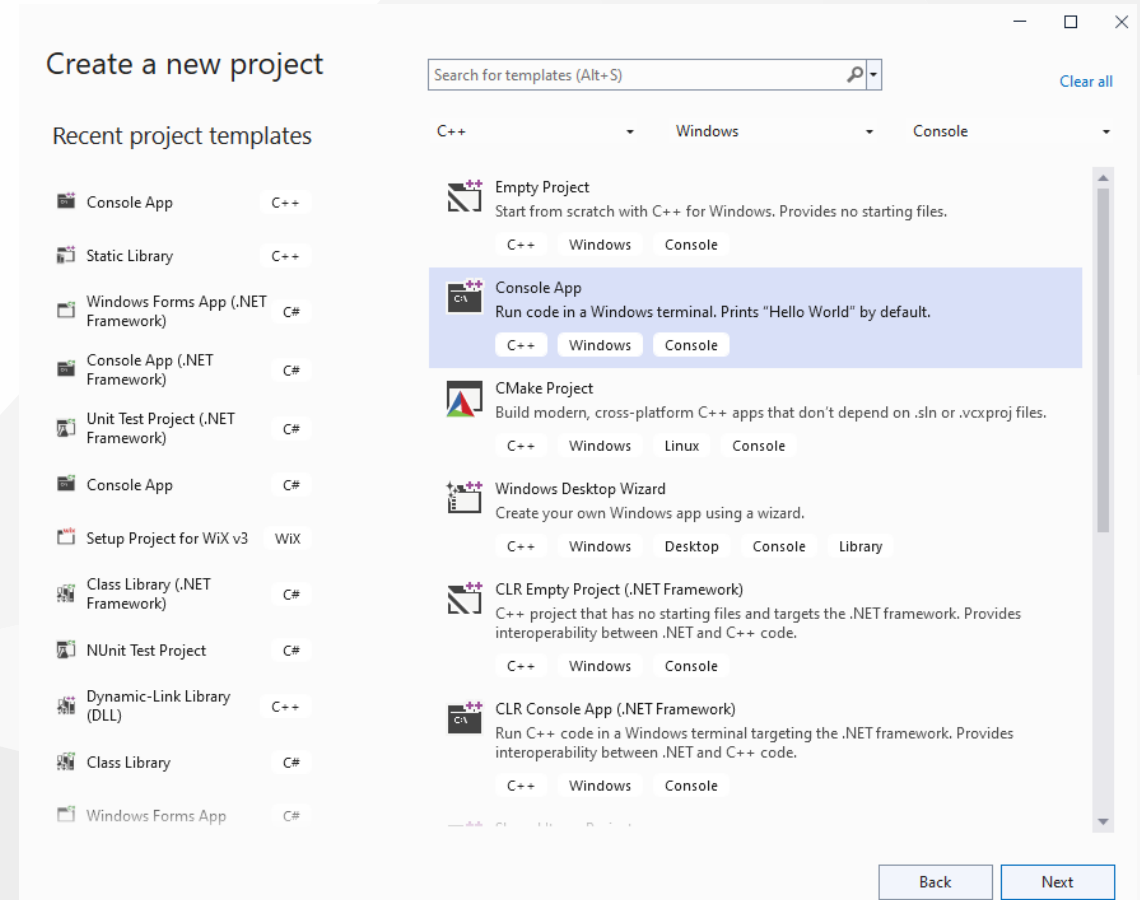
## Visual Studio Community Edition (Install / Compile / Run / Debug) (4)

- From Opening Window Select **Create a new project**



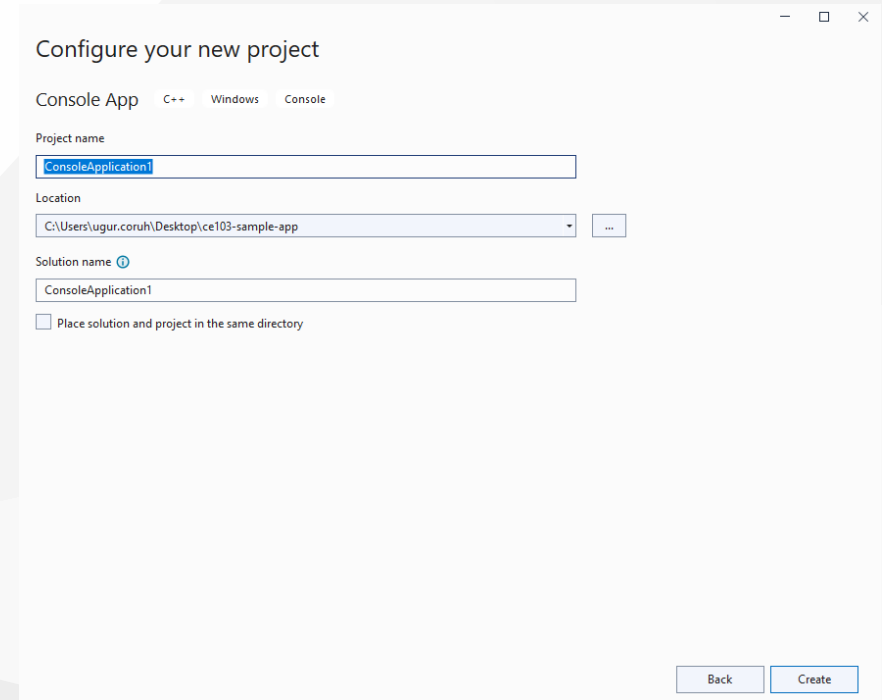
## Visual Studio Community Edition (Install / Compile / Run / Debug) (5)

- There will be several options, you can review them.
- Select **Windows** , **C++** , **Console Application** from Combobox.
- Select **Console Application**



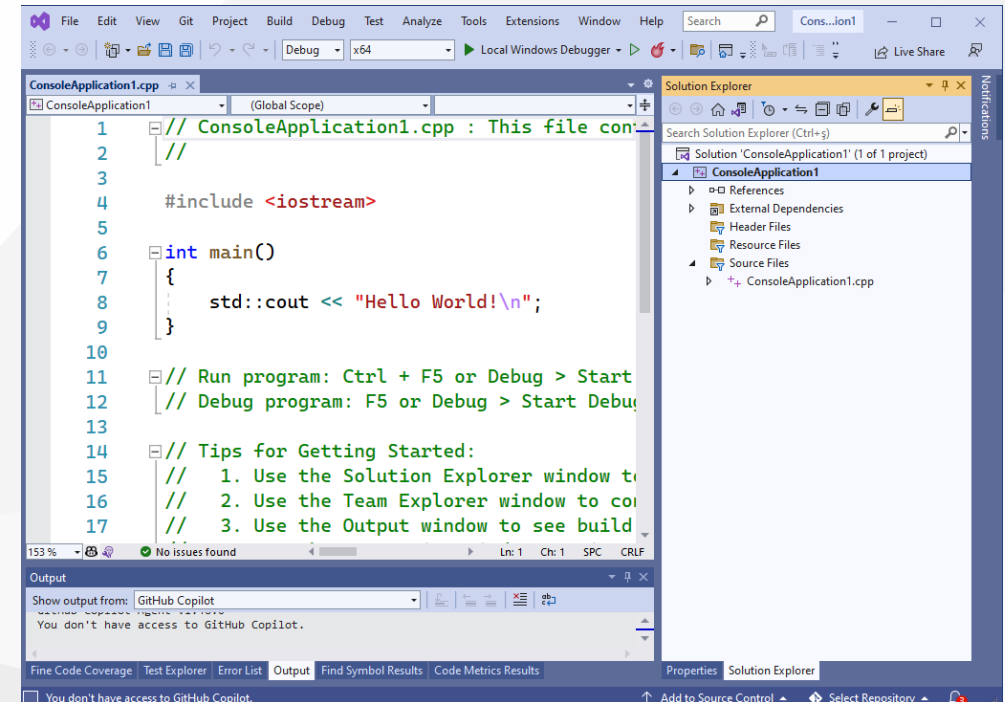
## Visual Studio Community Edition (Install / Compile / Run / Debug) (6)

- Give a solution and project name.
- Select save location



## Visual Studio Community Edition (Install / Compile / Run / Debug) (7)

- You will have C++ basic Hello World application.



The screenshot displays the Visual Studio Community Edition interface. The main editor window shows the source code for 'ConsoleApplication1.cpp'. The code includes the `<iostream>` header and a `main()` function that prints "Hello World!\n". The Solution Explorer on the right shows the project structure. The Output window at the bottom shows a message from GitHub Copilot: "You don't have access to GitHub Copilot." The status bar at the bottom indicates "No issues found" and "Ln: 1 Ch: 1 SPC CRLF".

```
1 // ConsoleApplication1.cpp : This file contains the source code for the application.
2 //
3
4 #include <iostream>
5
6 int main()
7 {
8     std::cout << "Hello World!\n";
9 }
10
11 // Run program: Ctrl + F5 or Debug > Start Without Debugging
12 // Debug program: F5 or Debug > Start Debugging
13
14 // Tips for Getting Started:
15 // 1. Use the Solution Explorer window to explore sources of the project.
16 // 2. Use the Team Explorer window to connect to remote sources.
17 // 3. Use the Output window to see build messages.
```

## Visual Studio Community Edition (Install / Compile / Run / Debug) (8)

- You will have C++ basic Hello World application.

```
// ConsoleApplication1.cpp : This file contains the 'main' function. Program execution begins and ends there.
//
#include <iostream>

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

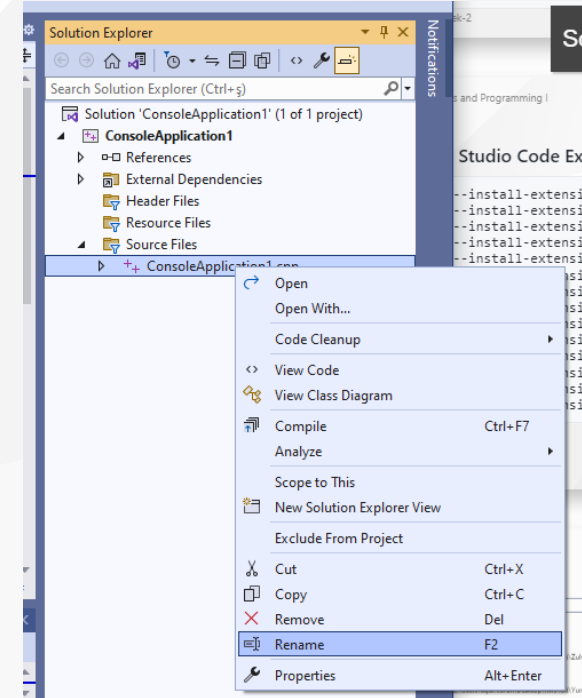
// Run program: Ctrl + F5 or Debug > Start Without Debugging menu
// Debug program: F5 or Debug > Start Debugging menu

// Tips for Getting Started:
// 1. Use the Solution Explorer window to add/manage files
// 2. Use the Team Explorer window to connect to source control
// 3. Use the Output window to see build output and other messages
// 4. Use the Error List window to view errors
// 5. Go to Project > Add New Item to create new code files, or Project > Add Existing Item to add existing code files to the project
// 6. In the future, to open this project again, go to File > Open > Project and select the .sln file
```



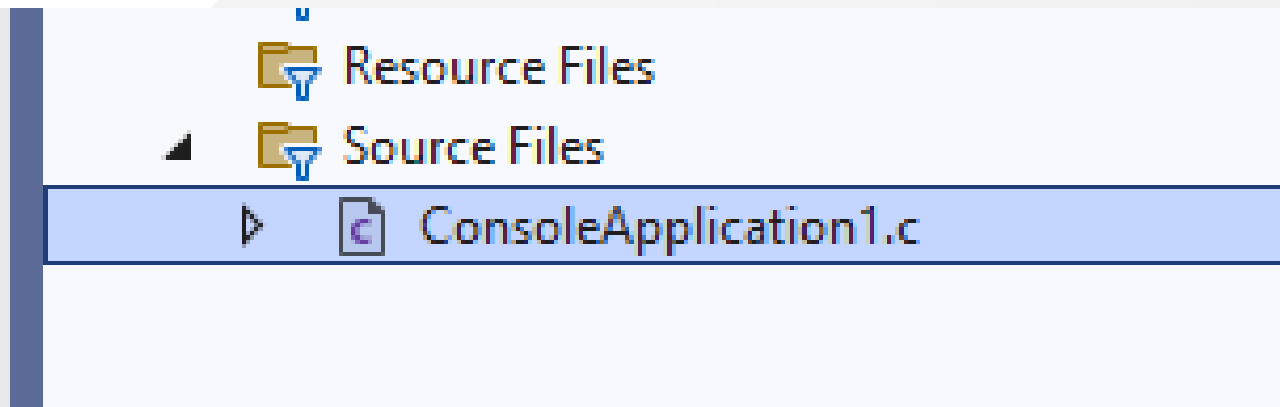
## Visual Studio Community Edition (Install / Compile / Run / Debug) (9)

- We need to rename the file extension to `c` from `cpp`



## Visual Studio Community Edition (Install / Compile / Run / Debug) (10)

- If you compile the source C compiler will be used.



## Visual Studio Community Edition (Install / Compile / Run / Debug) (11)

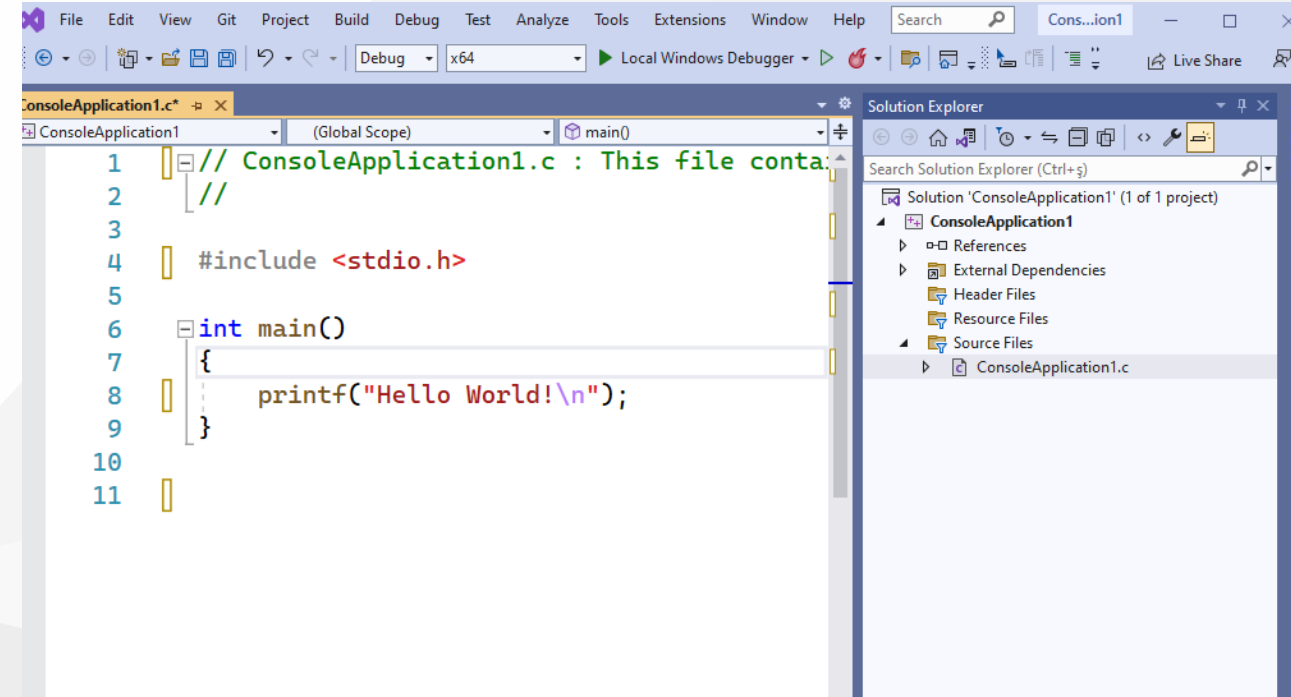
- We need to update our source for `c` as follows

```
// ConsoleApplication1.c : This file contains the 'main' function. Program execution begins and ends there.
//
#include <stdio.h>

int main(){
    printf("Hello World!\n");
}
```

## Visual Studio Community Edition (Install / Compile / Run / Debug) (12)

- We need to update our source for `C` as follows



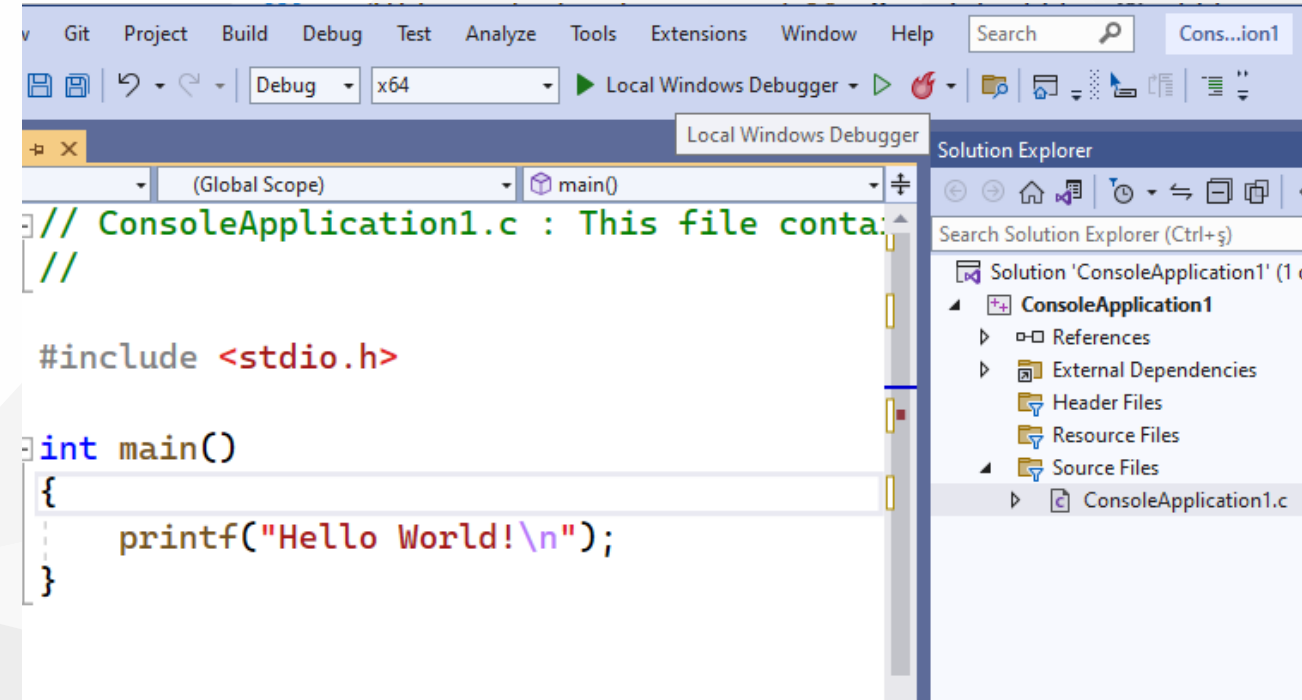
The screenshot shows the Visual Studio IDE with a C program in a console application. The code is as follows:

```
1 // ConsoleApplication1.c : This file contains the source code for the console application.
2 //
3
4 #include <stdio.h>
5
6 int main()
7 {
8     printf("Hello World!\n");
9 }
10
11
```

The Solution Explorer on the right shows the project structure for 'ConsoleApplication1' (1 of 1 project), including folders for References, External Dependencies, Header Files, Resource Files, and Source Files, with 'ConsoleApplication1.c' listed under Source Files.

## Visual Studio Community Edition (Install / Compile / Run / Debug) (13)

- Put a breakpoint by clicking the following location. Breakpoints will be stop points for our debugging operations.



The screenshot shows the Visual Studio IDE with the Local Windows Debugger active. The code editor displays the following C code:

```
// ConsoleApplication1.c : This file contains the source code for the console application.
//

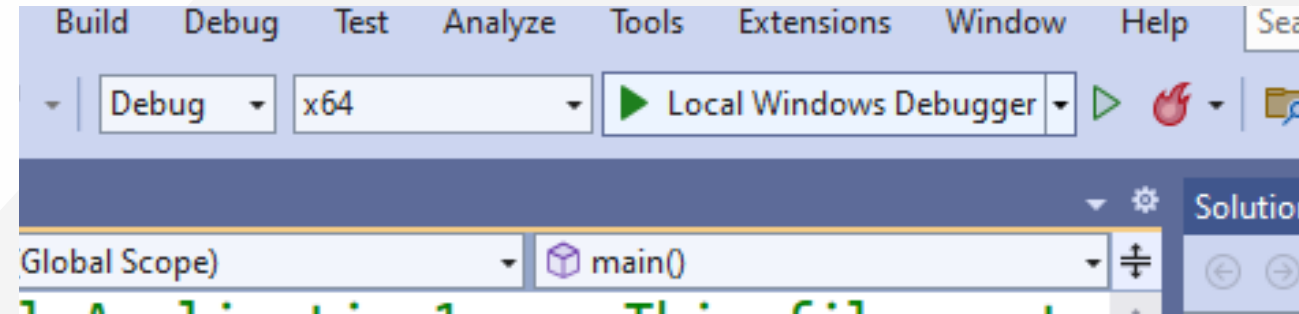
#include <stdio.h>

int main()
{
    printf("Hello World!\n");
}
```

A red vertical bar is placed on the left margin of the code editor, indicating a breakpoint is set on the line containing the opening curly brace of the `main()` function. The Solution Explorer on the right shows the project structure for 'ConsoleApplication1'.

## Visual Studio Community Edition (Install / Compile / Run / Debug) (14)

- Then select Debug configuration and according to your operating system select x64 or x86 configuration and click `Local Windows Debugger`



## Visual Studio Community Edition (Install / Compile / Run / Debug) (15)

- Update your source code as follow

```
// ConsoleApplication1.c : This file contains the 'main' function. Program execution begins and ends there.
//
#include <stdio.h>

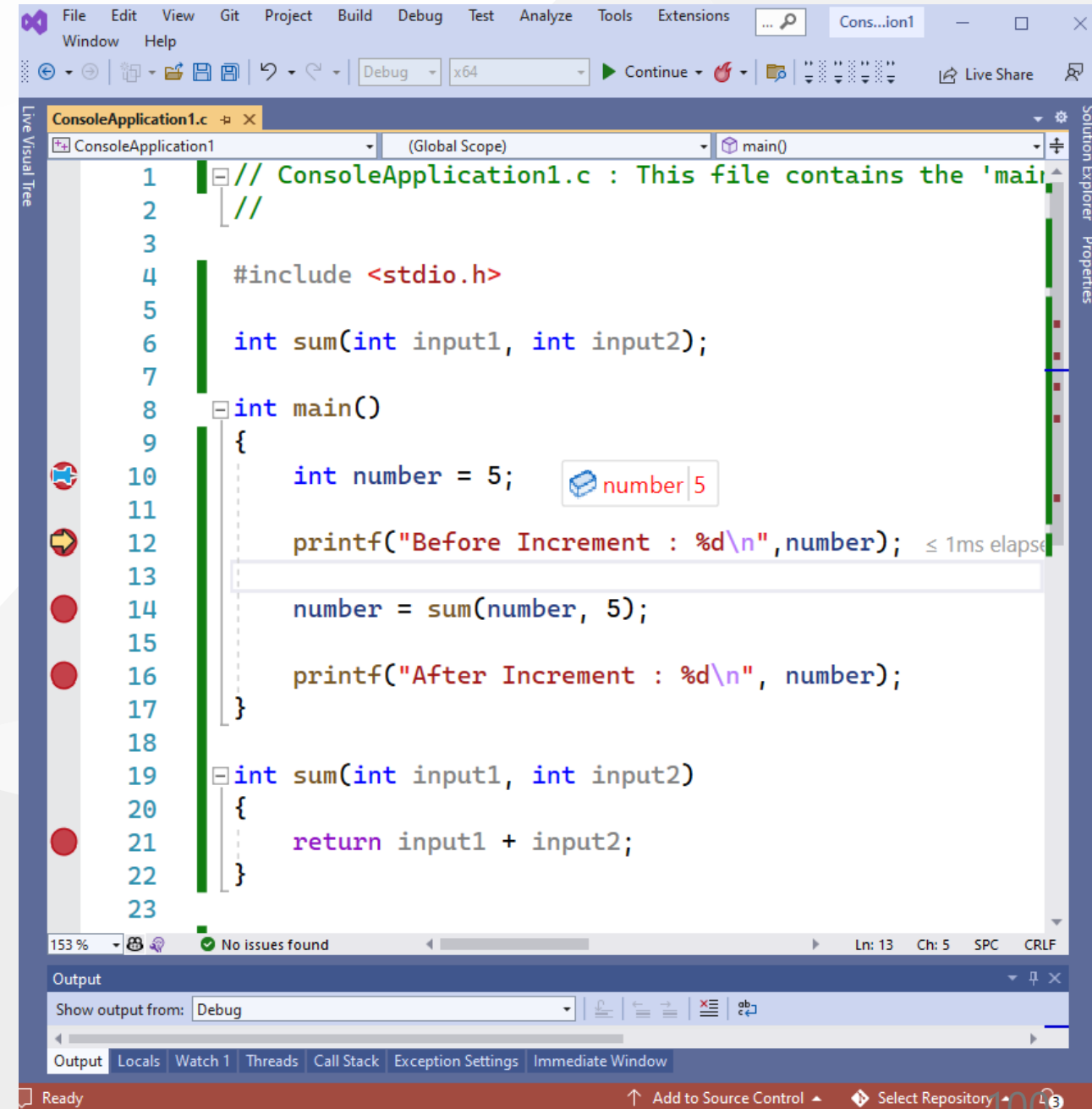
int sum(int input1, int input2);

int main(){
    int number = 5;
    printf("Before Increment : %d\n",number);
    number = sum(number, 5);
    printf("After Increment : %d\n", number);
}

int sum(int input1, int input2){
    return input1 + input2;
}
```

## Visual Studio Community Edition (Install / Compile / Run / Debug) (16)

- Put the following breakpoints and run the debugger. On number, the variable pins the variable to see its value in real-time.



The screenshot shows the Visual Studio IDE with a C program named 'ConsoleApplication1.c'. The code is as follows:

```
1 // ConsoleApplication1.c : This file contains the 'main' function.
2 //
3
4 #include <stdio.h>
5
6 int sum(int input1, int input2);
7
8 int main()
9 {
10     int number = 5;
11     printf("Before Increment : %d\n", number);
12     number = sum(number, 5);
13     printf("After Increment : %d\n", number);
14 }
15
16 int sum(int input1, int input2)
17 {
18     return input1 + input2;
19 }
20
21
22
23
```

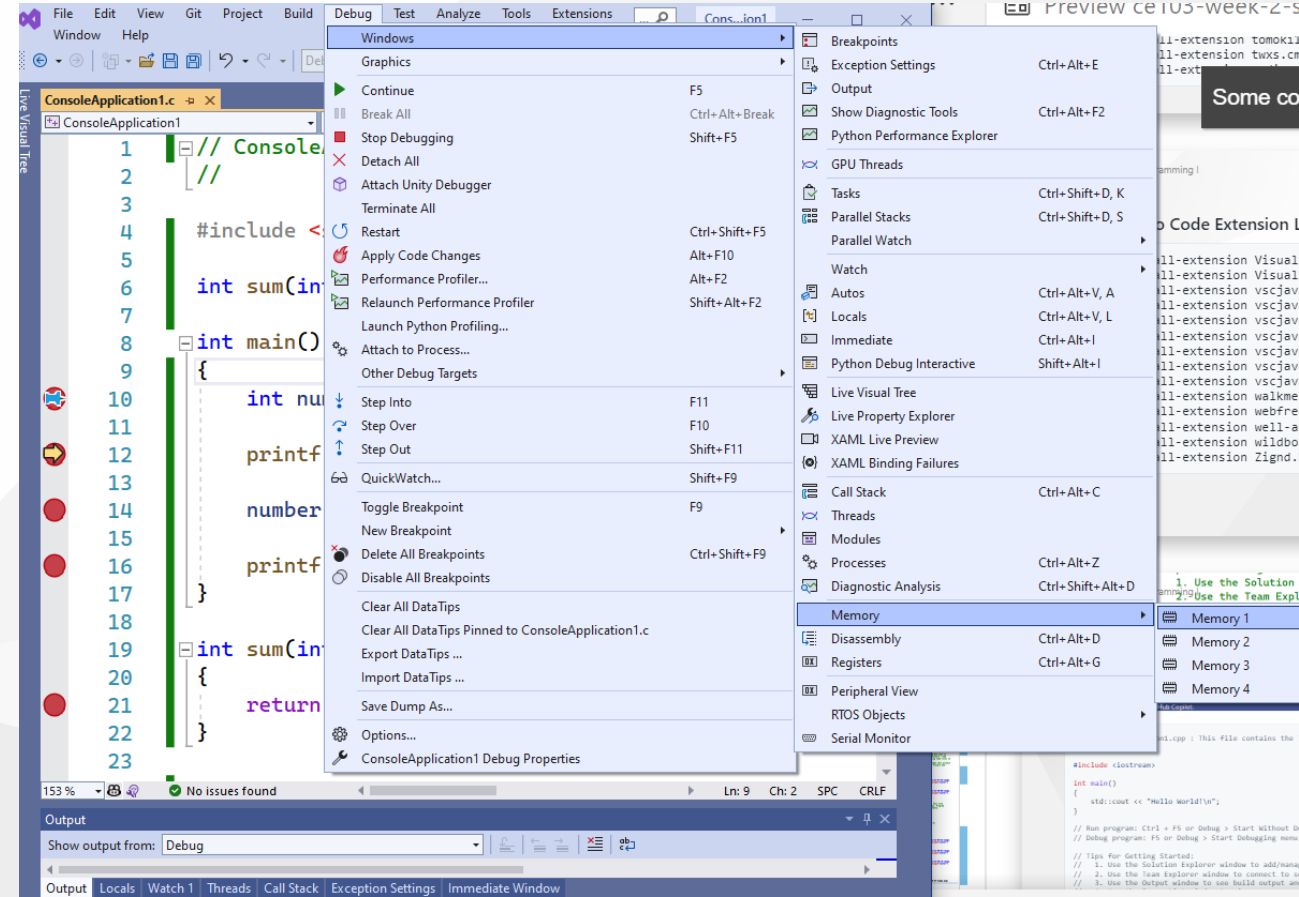
Breakpoints are set at lines 10, 12, 14, 16, and 19. A variable watch is set for 'number' at line 10, showing its value as 5. The output window shows the following text:

```
Before Increment : 5
After Increment : 10
```



## Visual Studio Community Edition (Install / Compile / Run / Debug) (17)

- Open Debug->Windows->Memory->Memory1 to see value in memory



## Visual Studio Community Edition (Install / Compile / Run / Debug) (18)

- In the memory window copy variable name (number) with address operator (&) and then (&number) press enter.

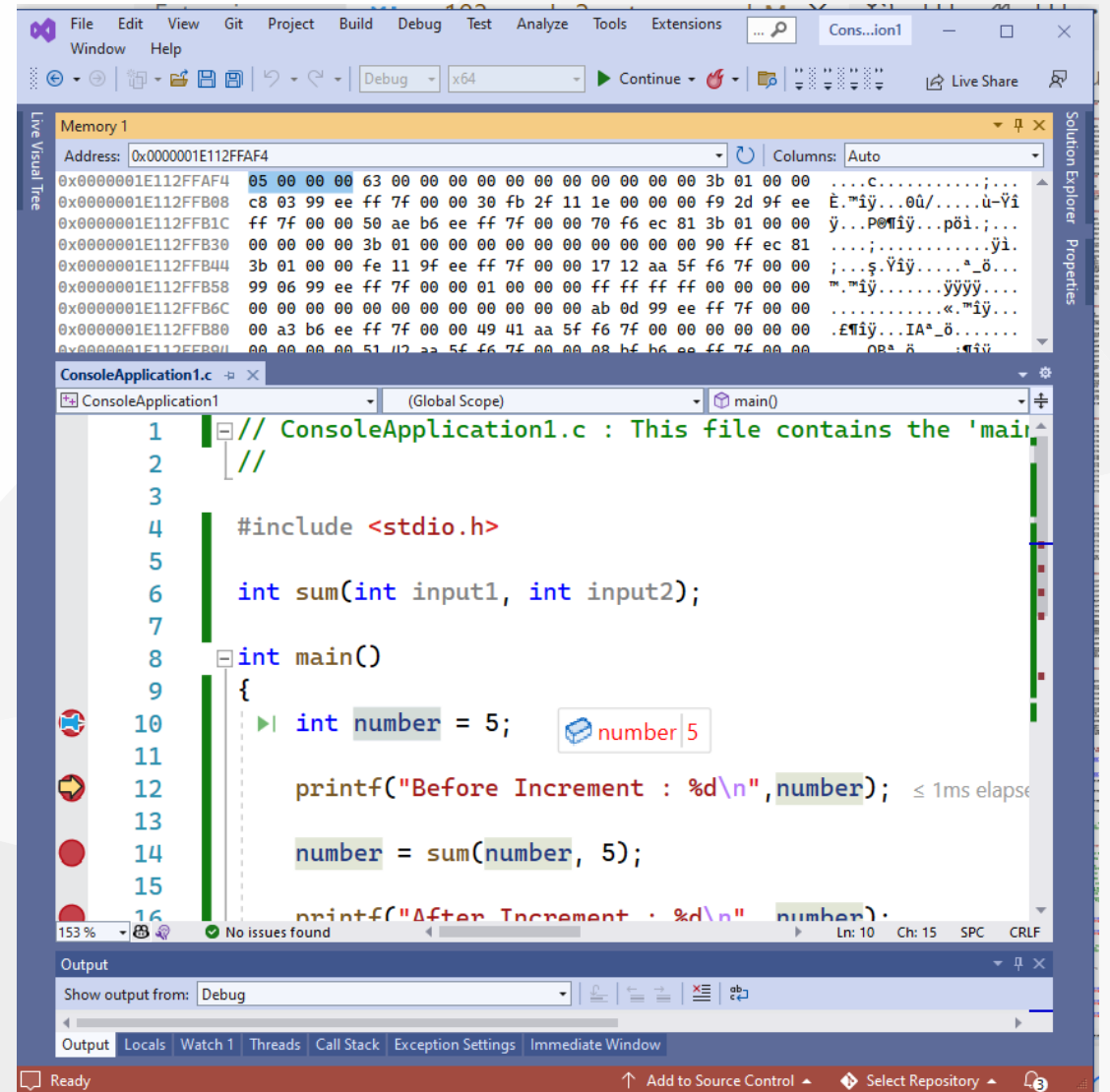
The screenshot shows the Visual Studio IDE in a debug state. The top toolbar includes a 'Continue' button and a 'Debug' dropdown menu. The 'Memory' window is open, showing a list of memory addresses and their corresponding values. The address field is set to '&number'. The source code window shows the following code:

```
1 // ConsoleApplication1.c : This file contain
2 //
3
4 #include <stdio.h>
5
6 int sum(int input1, int input2);
7
8 int main()
9 {
10     int number = 5;
11 }
```

The variable 'number' is highlighted in the code, and a tooltip shows its value as 5.

## Visual Studio Community Edition (Install / Compile / Run / Debug) (19)

- You can see its value in memory with hexadecimal form ( 05 00 00 00 )



## Visual Studio Community Edition (Install / Compile / Run / Debug) (20)

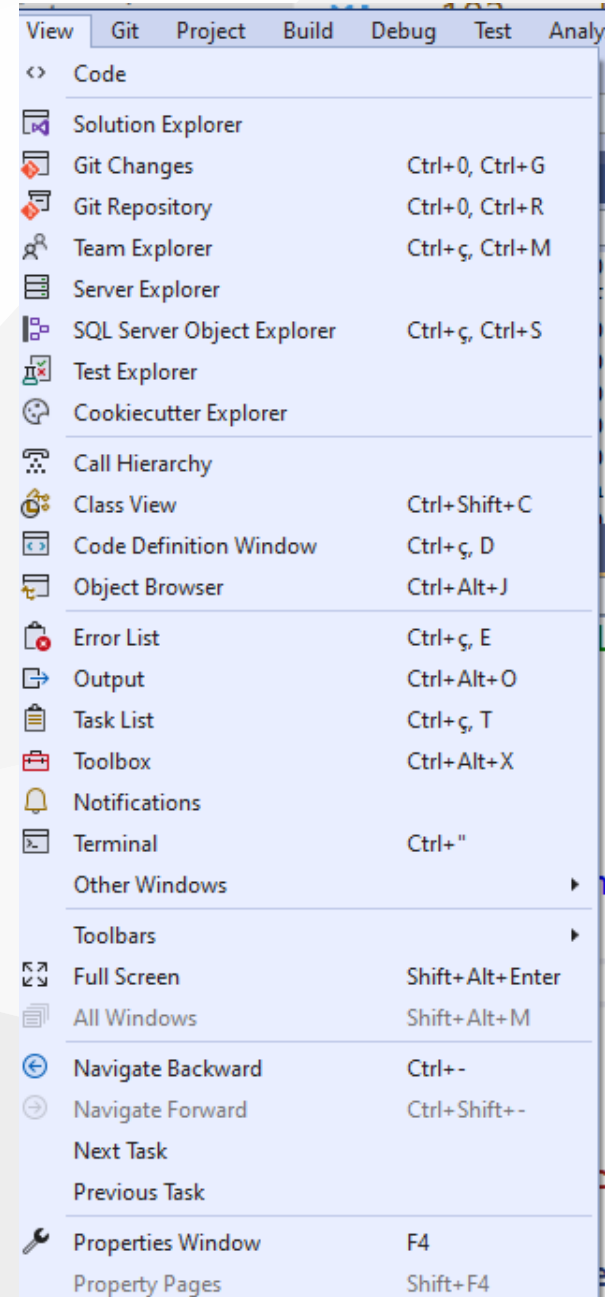
- If you change value with pinned control your memory value and your current value will be updated. 20 in hexadecimal 0x14 (integer is 4 bytes length for this reason memory shows 14 00 00 00 )

```
int main()
{
    int number = 5;
    printf("Before Increment : %d\n", number);
}
```

Memory 1	
Address:	0x0000001E112FFAF4
0x0000001E112FFAF4	14 00 00 00 63 00 00
0x0000001E112FFB08	c8 03 99 ee ff 7f 00
0x0000001E112FFB1C	ff 7f 00 00 50 ae b6
0x0000001E112FFB30	00 00 00 00 3b 01 00
0x0000001E112FFB44	3b 01 00 00 fe 11 9f

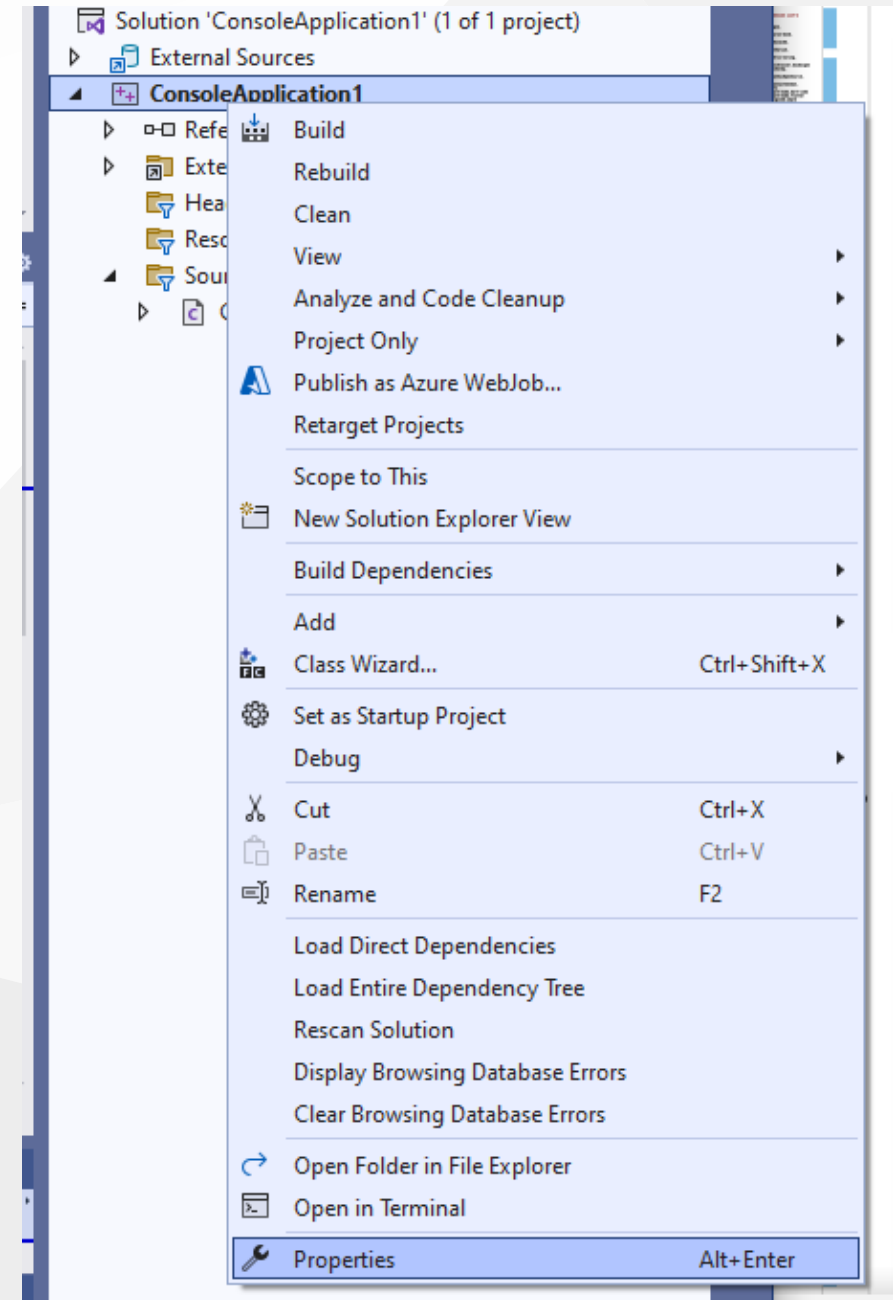
## Visual Studio Community Edition (Install / Compile / Run / Debug) (21)

- If you close some windows such as solution explorer, and properties windows you can open them from the View menu.



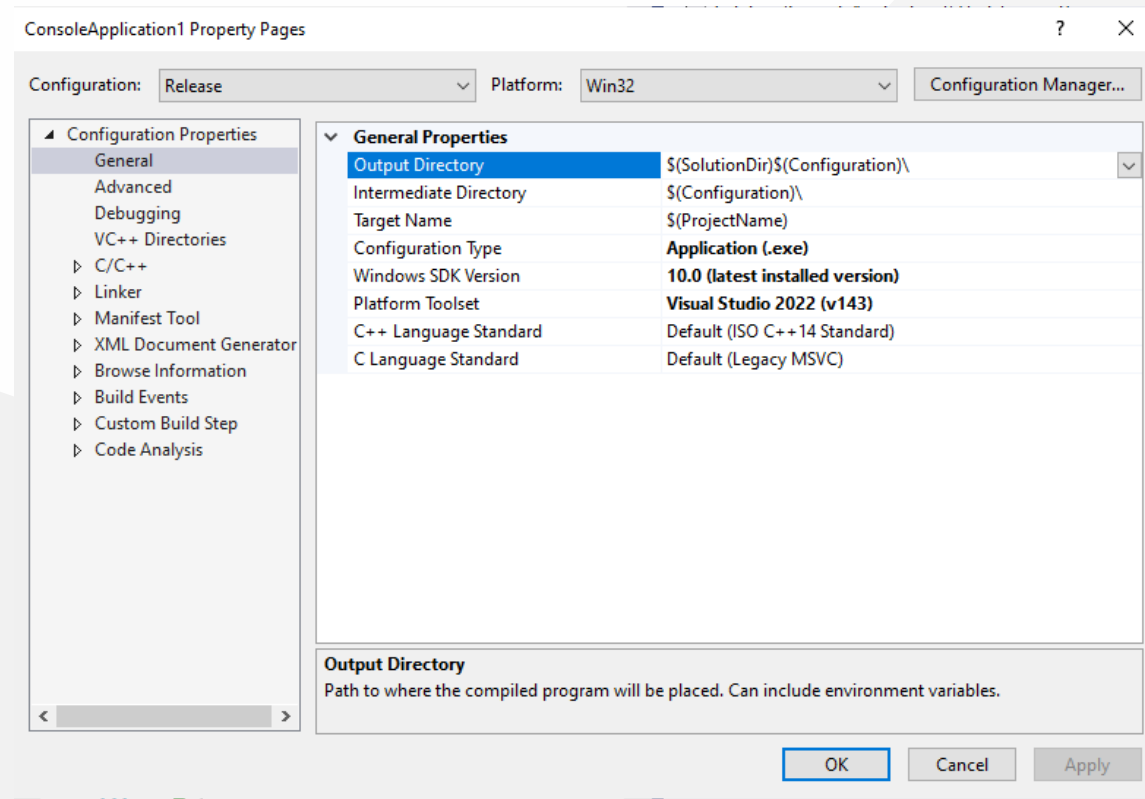
## Visual Studio Community Edition (Install / Compile / Run / Debug) (22)

- Solution and Projects have several configurations for each setup such as Release - x86, Release-x64, Debug-x86, and Debug-x64 you need to configure all of them for your requirements. You can access configurations by right-clicking to project and then selecting properties.

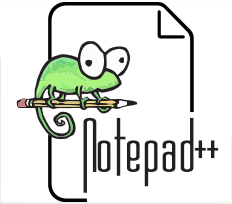


## Visual Studio Community Edition (Install / Compile / Run / Debug) (23)

- Project properties has several settings



## Notepad++ (Install / Compile ) (1)



- Please download Notepad++ from the following link
  - [Downloads | Notepad++](#)



## Notepad++ (Install / Compile ) (2)

Download and install MinGW or LLVM compiler (if you downloaded then skip this step)

### MinGW installer (gcc / g++)

- A complete runtime environment for gcc
  - <https://sourceforge.net/projects/mingw-w64/>
  - <https://sourceforge.net/projects/mingw-w64/files/latest/download>
- w64devkit is a portable C and C++ development kit for x64 (and x86) Windows.
  - <https://www.mingw-w64.org/downloads/#w64devkit>
- Also, see the following notes
  - <https://www.hanshq.net/building-gcc.html>

## Notepad++ (Install / Compile ) (3)

### LLVM installer (clang)

- Download
  - <https://releases.llvm.org/>
- Also, use the following notes
  - <https://llvm.org/devmtg/2014-04/PDFs/Talks/clang-cl.pdf>
  - <https://www.hanshq.net/clang-plugin-example.html>

## Notepad++ (Install / Compile ) (4)

Open a console with " `cmd` " and test the following commands if commands are not recognized then set the system environment variable to add `gcc` and `g++` executable paths to the path variable (add to both system and user path variable)

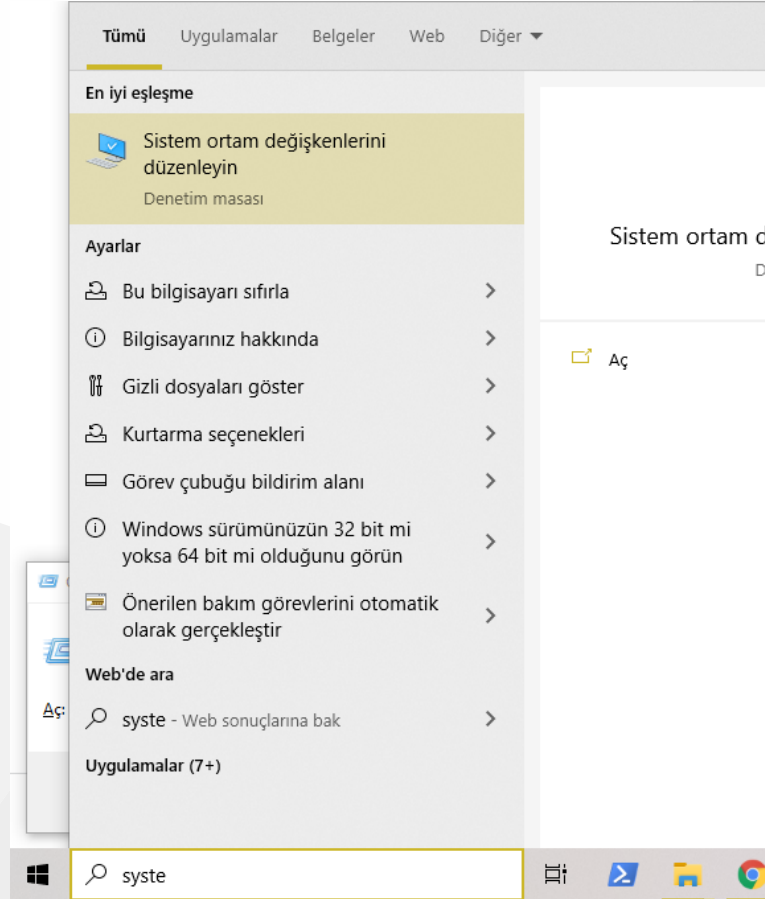
```
gcc --version
```

```
g++ --version
```

```
C:\Users\ugur.coruh>gcc --version
gcc (x86_64-win32-seh-rev0, Built by MinGW-W64 project) 8.1.0
Copyright (C) 2018 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
```

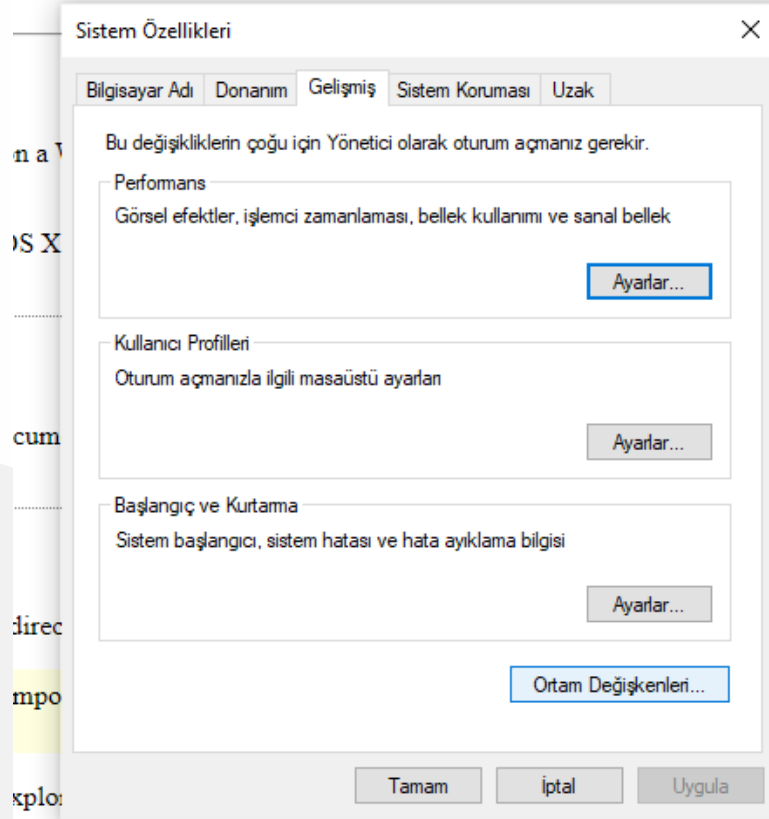
## Notepad++ (Install / Compile ) (5)

- Open system environments to update path variable for gcc/g++ and clang



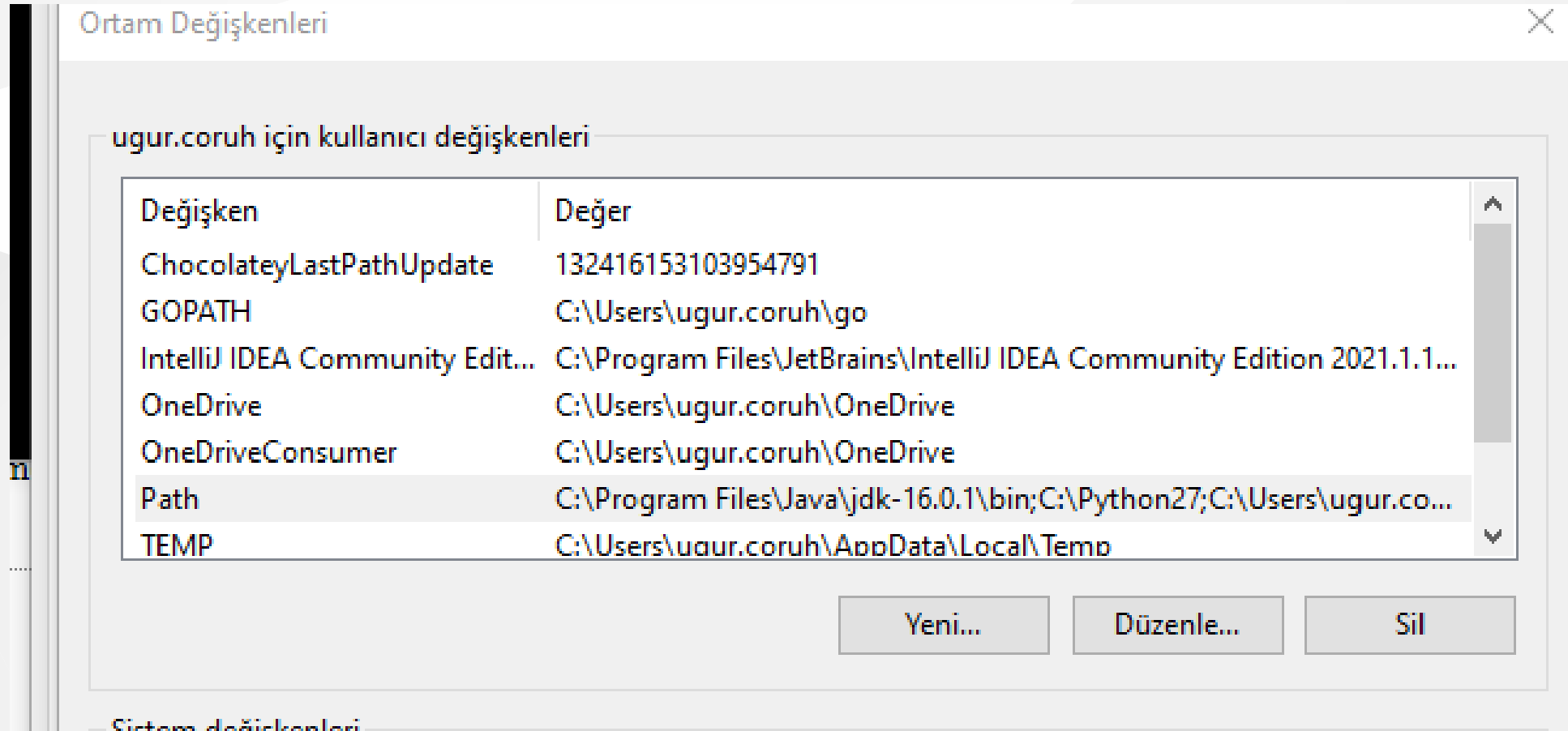
## Notepad++ (Install / Compile ) (6)

- Open "Environment Variables"



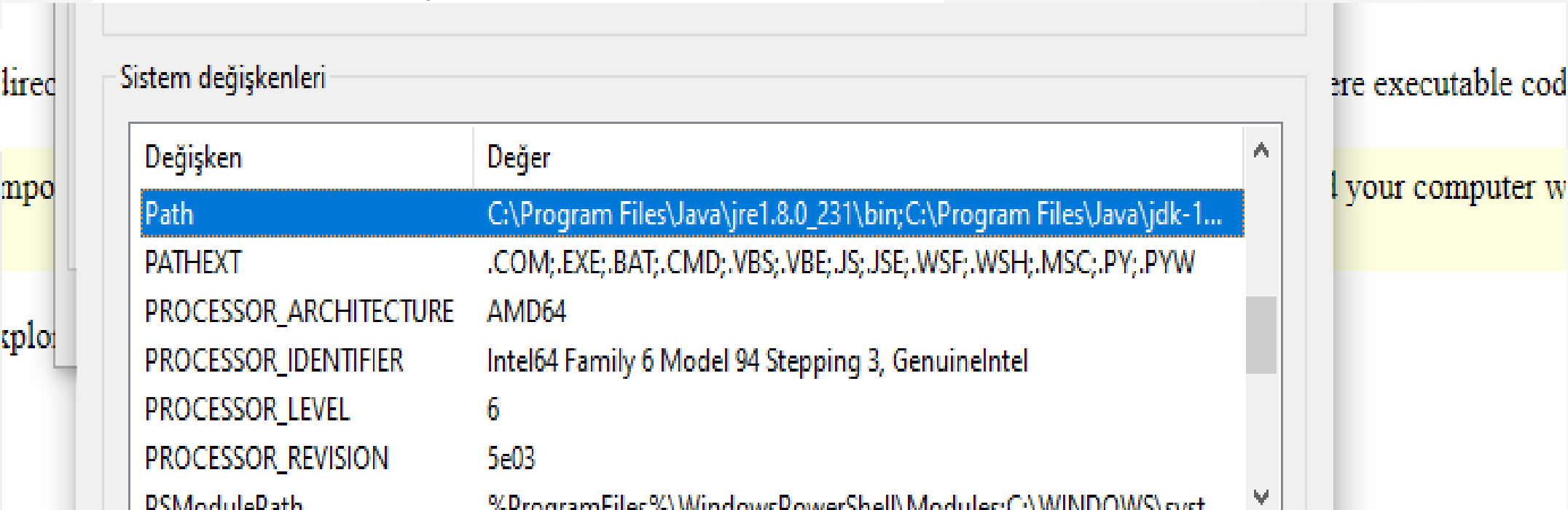
## Notepad++ (Install / Compile ) (7)

- Select path variable from user section.



## Notepad++ (Install / Compile ) (8)

- Select path variable from system section.

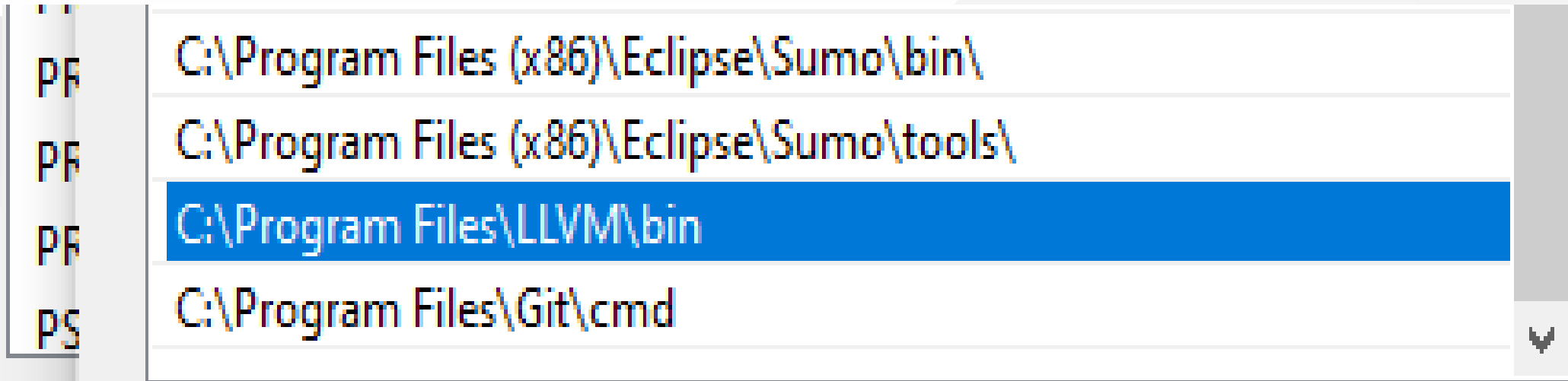


The screenshot shows the 'Sistem değişkenleri' (System Variables) dialog box in Windows. The 'Path' variable is selected and highlighted in blue. The value of the Path variable is 'C:\Program Files\Java\jre1.8.0\_231\bin;C:\Program Files\Java\jdk-1...'. Other variables listed include PATHEXT, PROCESSOR\_ARCHITECTURE, PROCESSOR\_IDENTIFIER, PROCESSOR\_LEVEL, PROCESSOR\_REVISION, and PSModulePath.

Değişken	Değer
Path	C:\Program Files\Java\jre1.8.0_231\bin;C:\Program Files\Java\jdk-1...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;.JSE;.WSF;.WSH;.MSC;.PY;.PYW
PROCESSOR_ARCHITECTURE	AMD64
PROCESSOR_IDENTIFIER	Intel64 Family 6 Model 94 Stepping 3, GenuineIntel
PROCESSOR_LEVEL	6
PROCESSOR_REVISION	5e03
PSModulePath	%ProgramFiles%\WindowsPowerShell\Modules;C:\WINDOWS\sys...

## Notepad++ (Install / Compile ) (9)

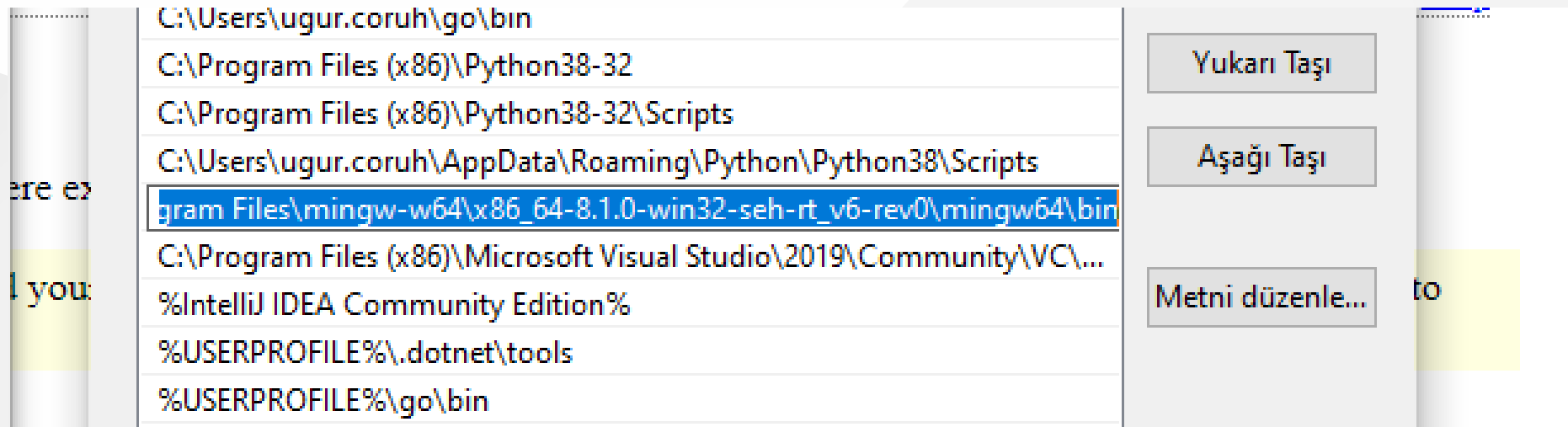
- Update variables add `MinGW` and `LLVM` to path `gcc.exe` `g++.exe` `clang.exe` will be in bin folders. Then we can run this commands from command line.





## Notepad++ (Install / Compile ) (9)

- Update variables add `MinGW` and `LLVM` to path `gcc.exe` `g++.exe` `clang.exe` will be in bin folders. Then we can run this commands from command line.



## Notepad++ (Install / Compile ) (10)

- for `gcc.exe` , `g++.exe` and `gdb.exe`

```
C:\Program Files\mingw-w64\x86_64-8.1.0-win32-seh-rt_v6-rev0\mingw64\bin
```

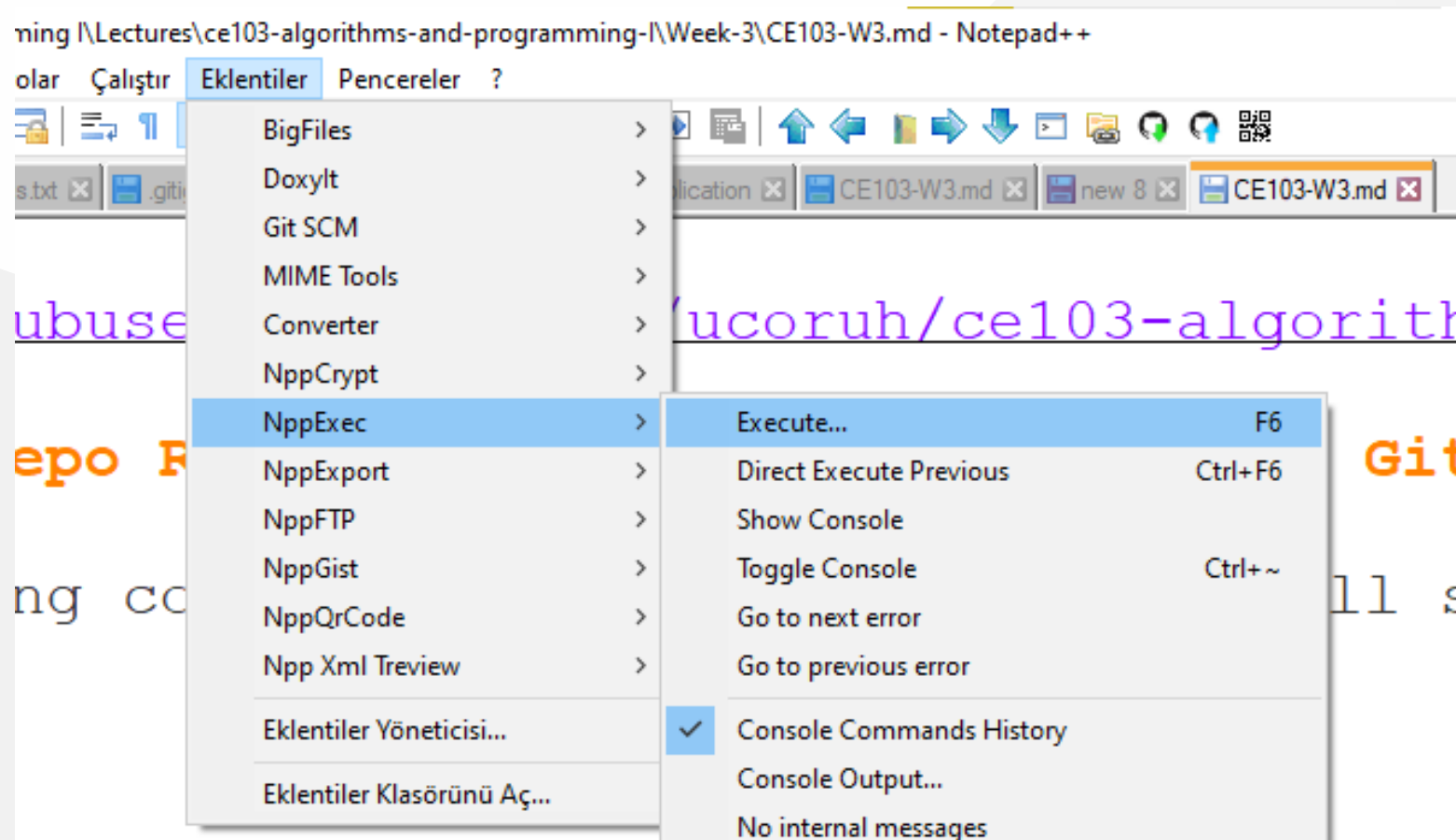
## Notepad++ (Install / Compile ) (11)

- for `clang.exe` , `lldb.exe` we will use the following path

```
C:\Program Files\LLVM\bin
```

## Notepad++ (Install / Compile ) (12)

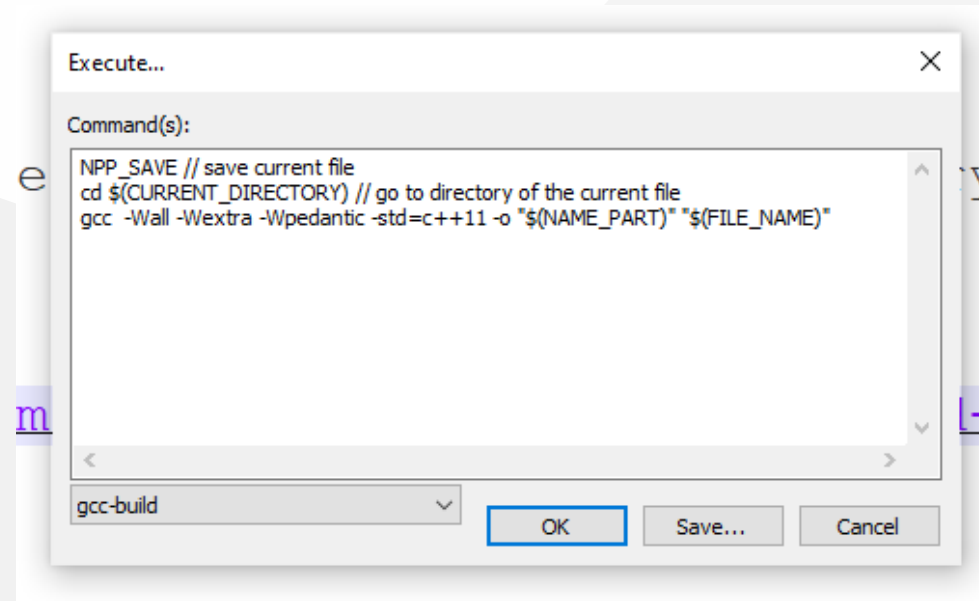
- This folder paths changes according to your setup
- Open `NppExec` extension (install from extension manager if not exist)



## Notepad++ (Install / Compile ) (13)

- Write the following commands in the box

```
NPP_SAVE // save current file  
cd $(CURRENT_DIRECTORY) // go to directory of the current file  
gcc -Wall -Wextra -Wpedantic -std=c++11 -o "$(NAME_PART)" "$(FILE_NAME)"
```



## Notepad++ (Install / Compile ) (14)

- Save the script as `gcc-build` and for more information check the following link
- You can modify or add multiple scripts for another task.

## MSYS2

- Software Distribution and Building Platform for Windows

<https://www.msys2.org/>

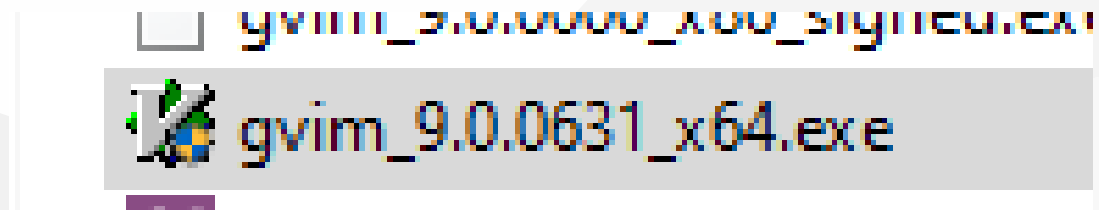
## Vi/Vim (C/C++) for Windows (1)

- Vim is a command-line editor for programming
- Use the following links to download Vim for Windows
  - <https://github.com/vim/vim-win32-installer/releases>
  - [download : vim online](#)

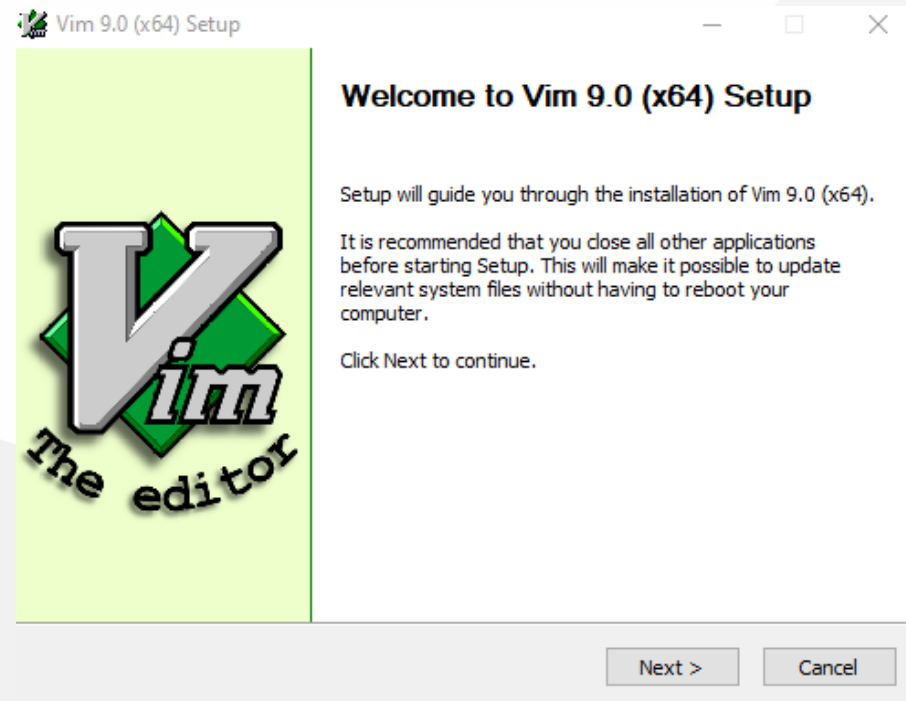


## Vi/Vim (C/C++) for Windows (2)

- Run setup to install the application on your computer.

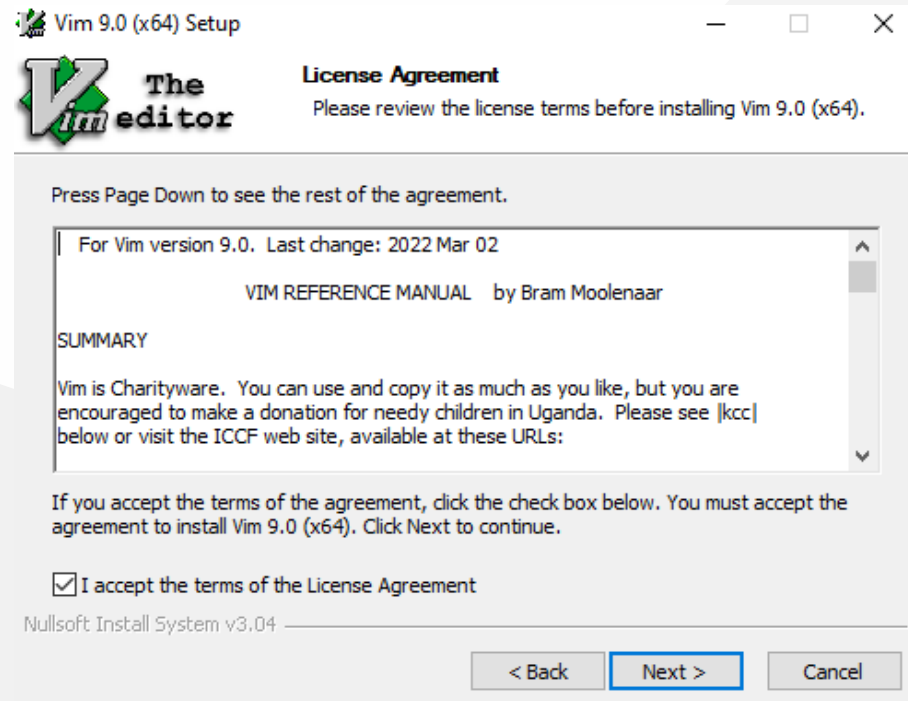


## Vi/Vim (C/C++) for Windows (3)



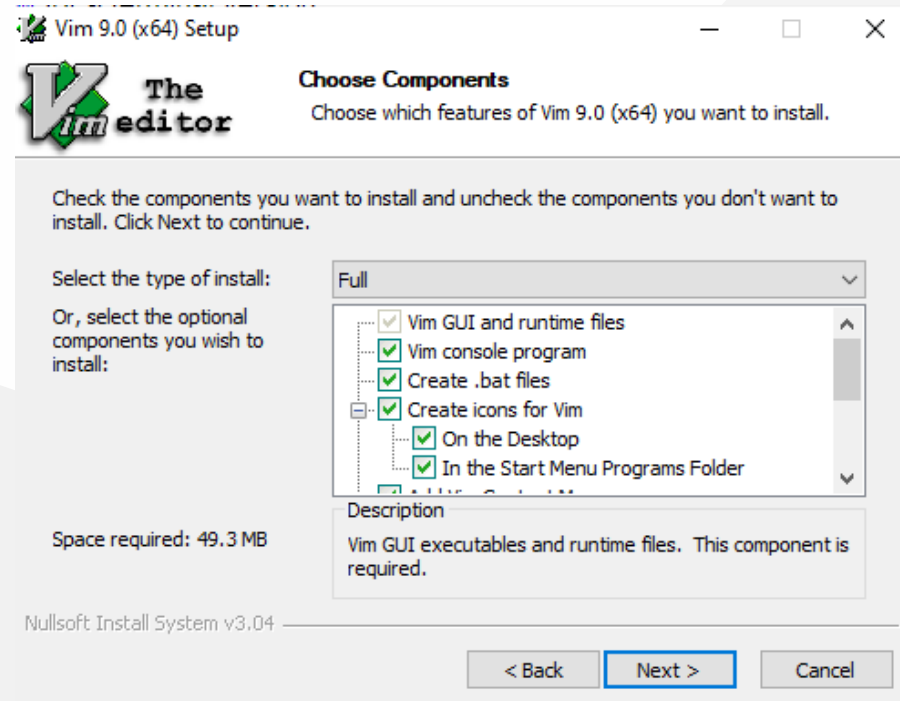
## Vi/Vim (C/C++ ) for Windows (4)

- Installation steps.



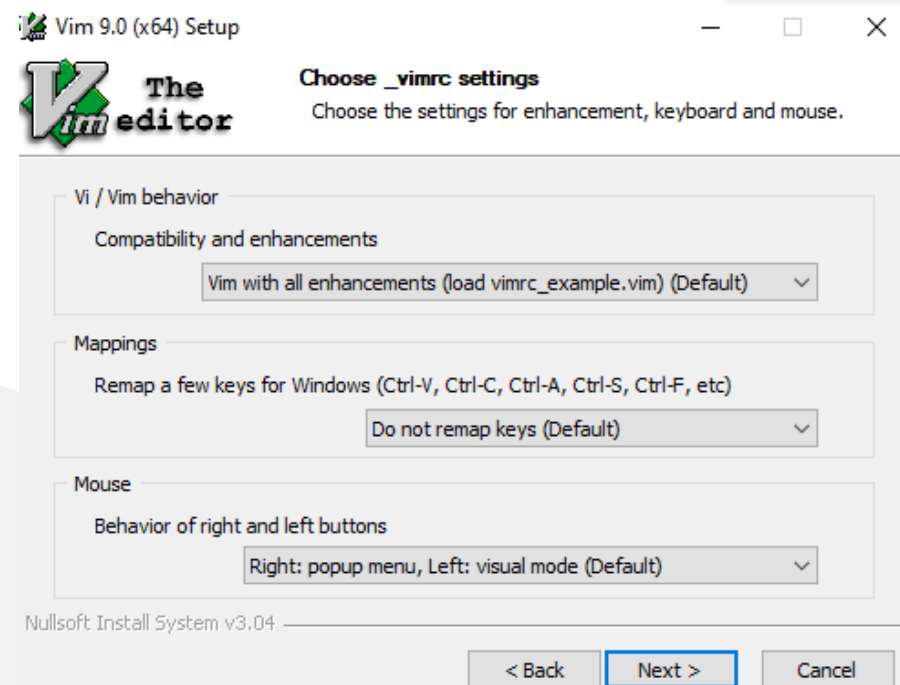
## Vi/Vim (C/C++ ) for Windows (5)

- Installation steps.



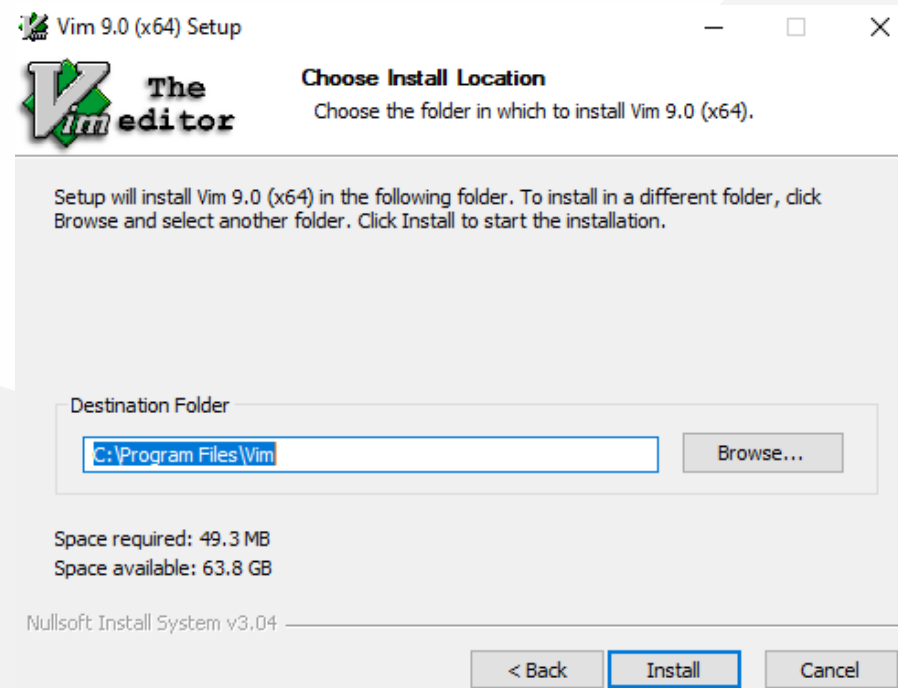
## Vi/Vim (C/C++) for Windows (6)

- Installation steps.



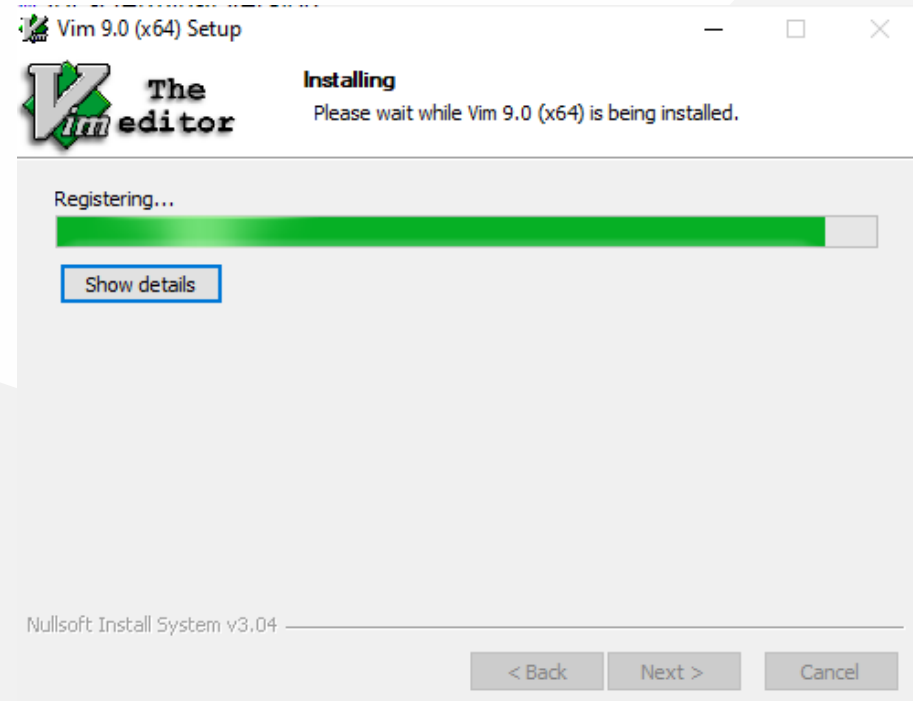
## Vi/Vim (C/C++) for Windows (7)

- Installation steps.



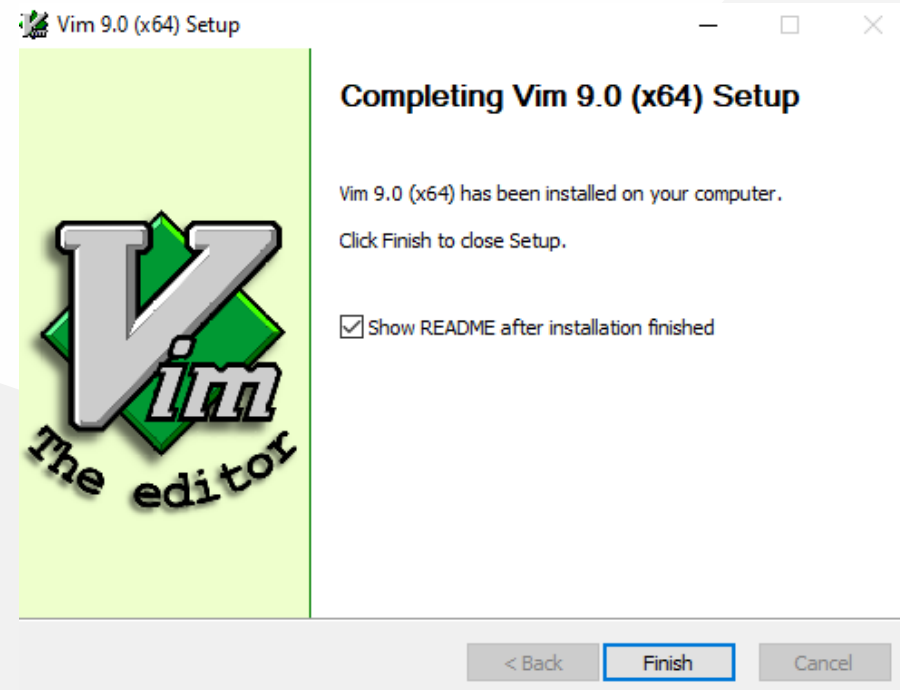
## Vi/Vim (C/C++) for Windows (8)

- Installation steps.



## Vi/Vim (C/C++) for Windows (9)

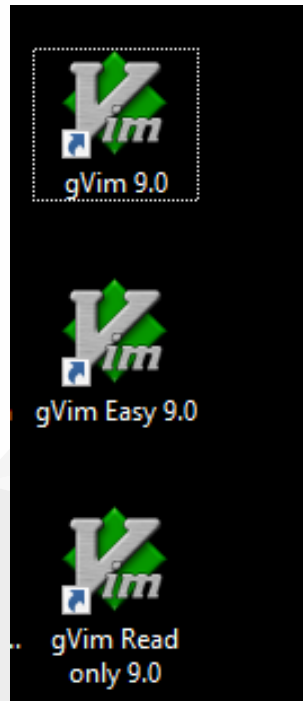
- Installation steps.





## Vi/Vim (C/C++) for Windows (10)

- Generated shortcuts on your desktop



## Vi/Vim (C/C++) for Windows (11)

- Run `vim hello.c` on your command-line to open a c file with vim editor.

```
WINDOWS\system32\cmd.exe
```

```
Microsoft Windows [Version 10.0.19044.2006]  
Copyright (c) 2019 Microsoft Corporation. All rights reserved.
```

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

```
C:\Users\ugur.coruh\Desktop>mkdir vim-sample-project
```

```
C:\Users\ugur.coruh\Desktop>cd vim-sample-project
```

```
C:\Users\ugur.coruh\Desktop\vim-sample-project>dir  
Volume in drive C is Windows  
Volume Serial Number is 8C3C-8F8C
```

```
Directory of C:\Users\ugur.coruh\Desktop\vim-sample-project
```

```
2022/08/15 15:36 <DIR> .  
2022/08/15 15:36 <DIR> ..  
0 File(s) 0 bytes  
2 Dir(s) 68.409.643.008 bytes free
```

```
C:\Users\ugur.coruh\Desktop\vim-sample-project>vim hello.c
```

## Vi/Vim (C/C++) for Windows (12)

- You will have the following editor.
- Use INSERT to change edit options.



The screenshot shows a Vim editor window titled "hello.c + (~\Desktop\vim-sample-project) - VIM1". The code displayed is a C program that prompts the user for their name and prints it back. The code is as follows:

```
#include <stdio.h>
int main()
{
    char name[20];
    printf("Enter Name:");
    scanf("%s",name);
    printf("Your name is %s",name);
    return 0;
}
```

At the bottom of the window, the status bar shows "-- EKLE --" on the left, "9,2" in the center, and "Tüm Belge" on the right.

## Vi/Vim (C/C++) for Windows (13)

- Sample source code

```
#include <stdio.h>
int main()
{
    char name[20];
    printf("Enter Name:");
    scanf("%s",name);
    printf("Your name is %s",name);
    return 0;
}
```

## Vi/Vim (C/C++) for Windows (14)

- Write source code
- Press the `Esc` button to enter command mode
- Then type `:wq` . It will save the file and exit from Vim
  - `w`: write
  - `q`: quit



```
hello.c + (~\Desktop\vim-sample-project) - VIM1
#include <stdio.h>
int main()
{
    char name[20];
    printf("Enter Name:");
    scanf("%s", name);
    printf("Your name is %s", name);
    return 0;
}
:wq
```

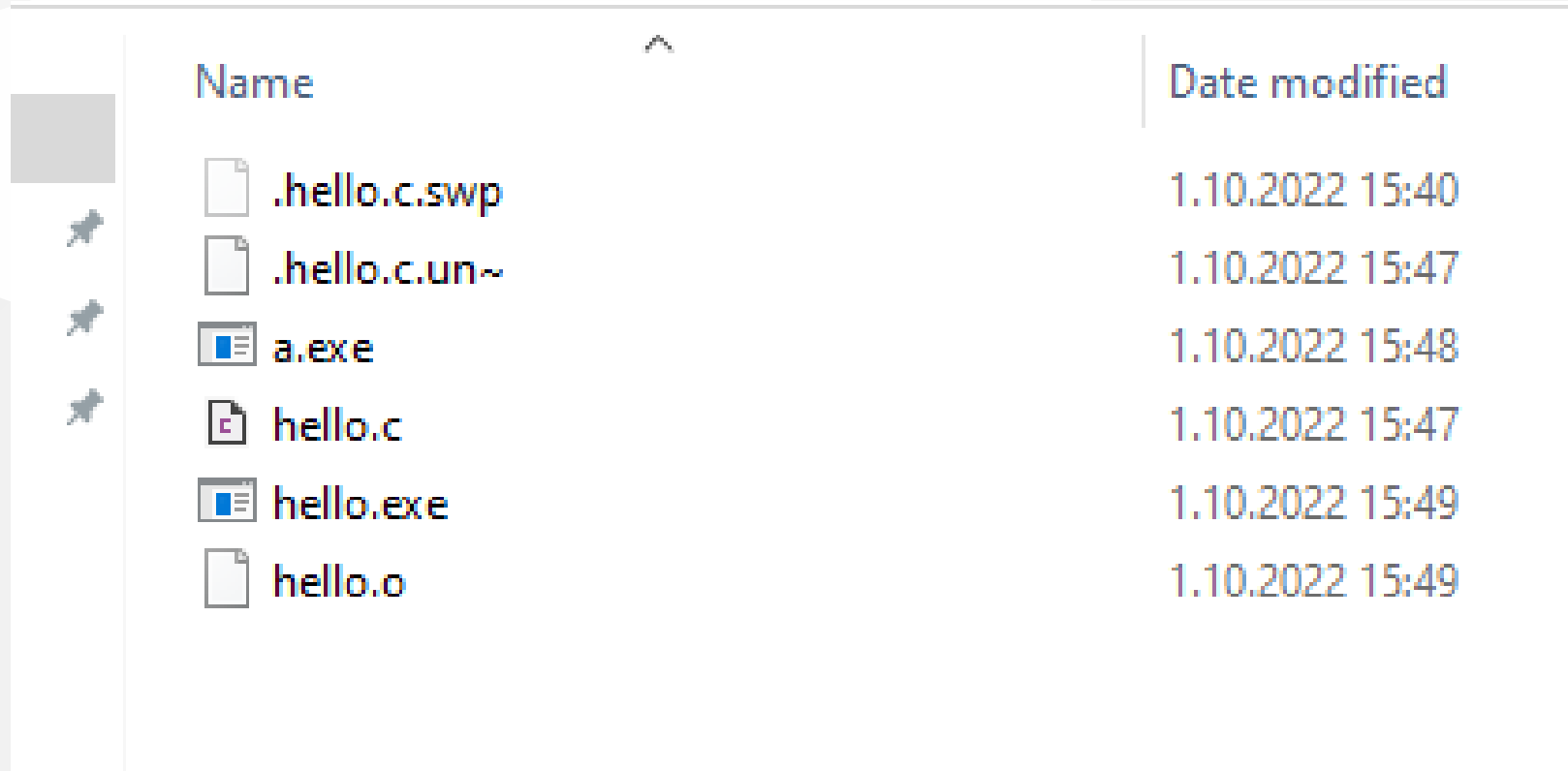
## Vi/Vim (C/C++ ) for Windows (15)







- compile source code with `gcc`
- link the objects and
- run executable

```
C:\Users\ugur.coruh\Desktop\vim-sample-project>gcc -c hello.c -o hello.o
C:\Users\ugur.coruh\Desktop\vim-sample-project>gcc hello.o -o hello.exe
C:\Users\ugur.coruh\Desktop\vim-sample-project>hello.exe
Enter Name:Ugur Coruh
Your name is Ugur
C:\Users\ugur.coruh\Desktop\vim-sample-project>
```

## Vi/Vim (C/C++) for Windows (17)

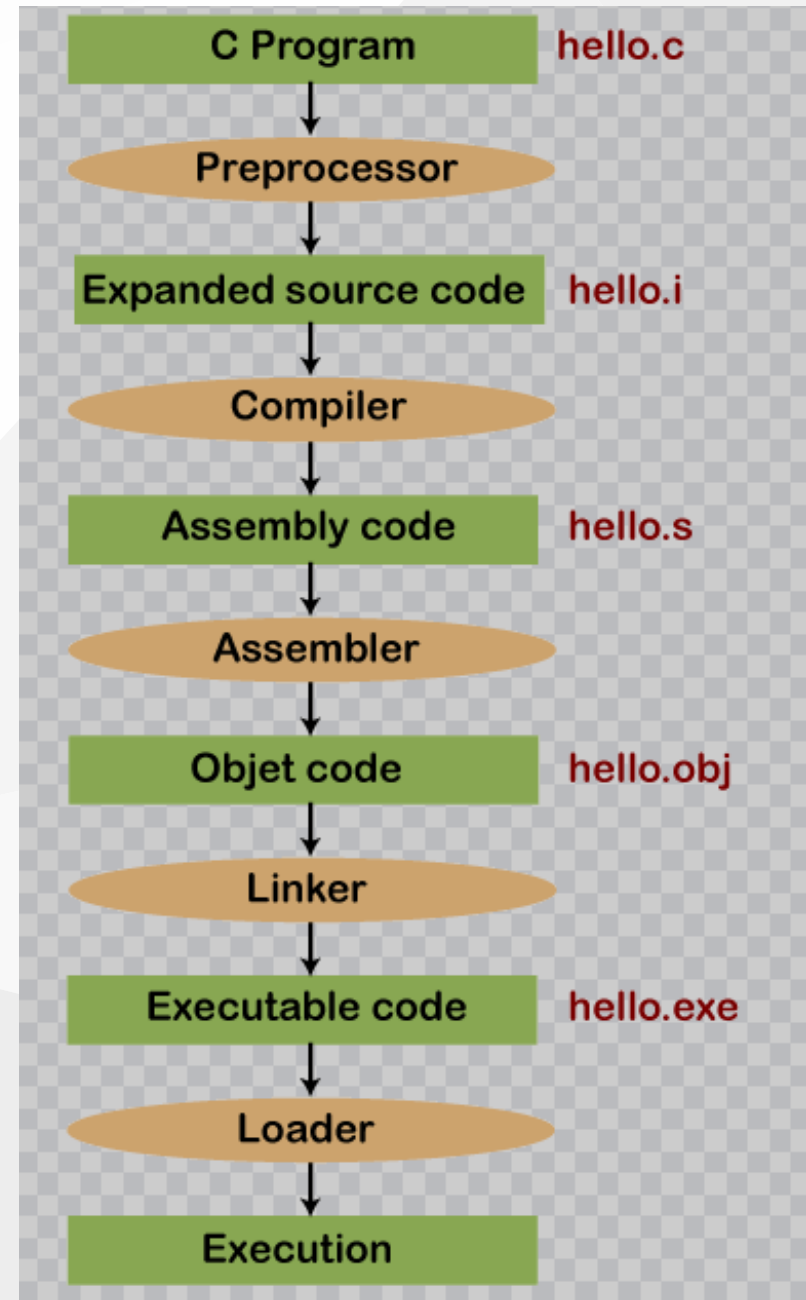
- In the folder, you can find your executable. `hello.exe`



Name	Date modified
 .hello.c.swp	1.10.2022 15:40
 .hello.c.un~	1.10.2022 15:47
 a.exe	1.10.2022 15:48
 hello.c	1.10.2022 15:47
 hello.exe	1.10.2022 15:49
 hello.o	1.10.2022 15:49

## Vi/Vim (C/C++) for Windows (16)

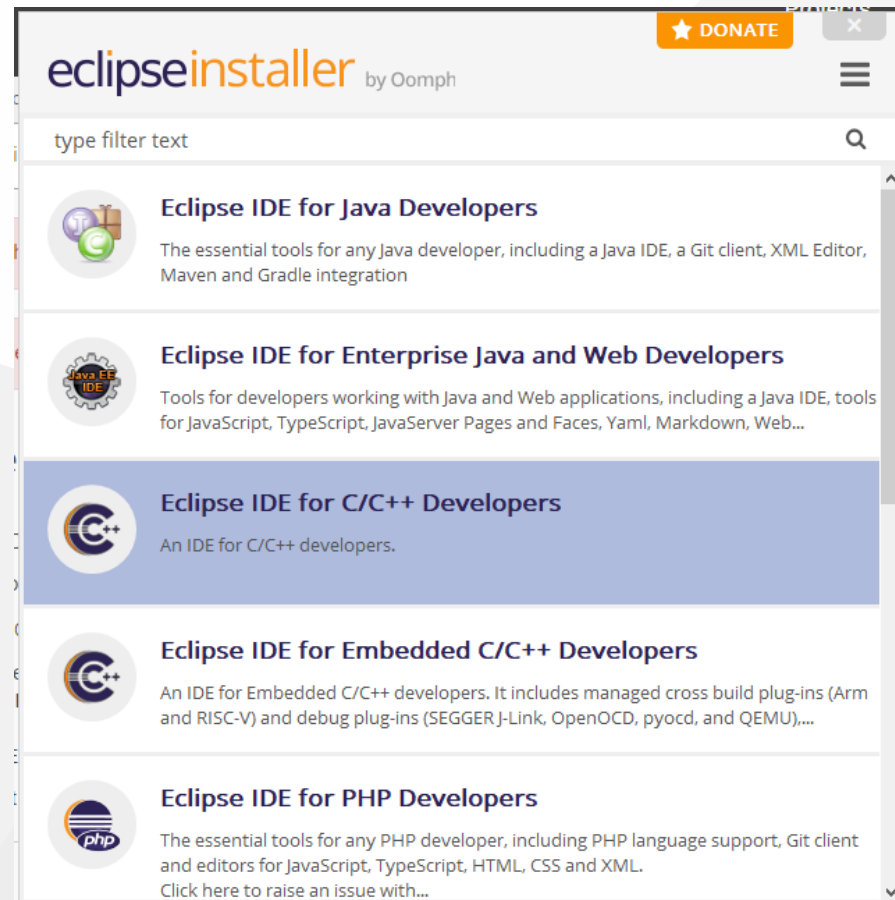
- compile, link and execute flow will be as follow;





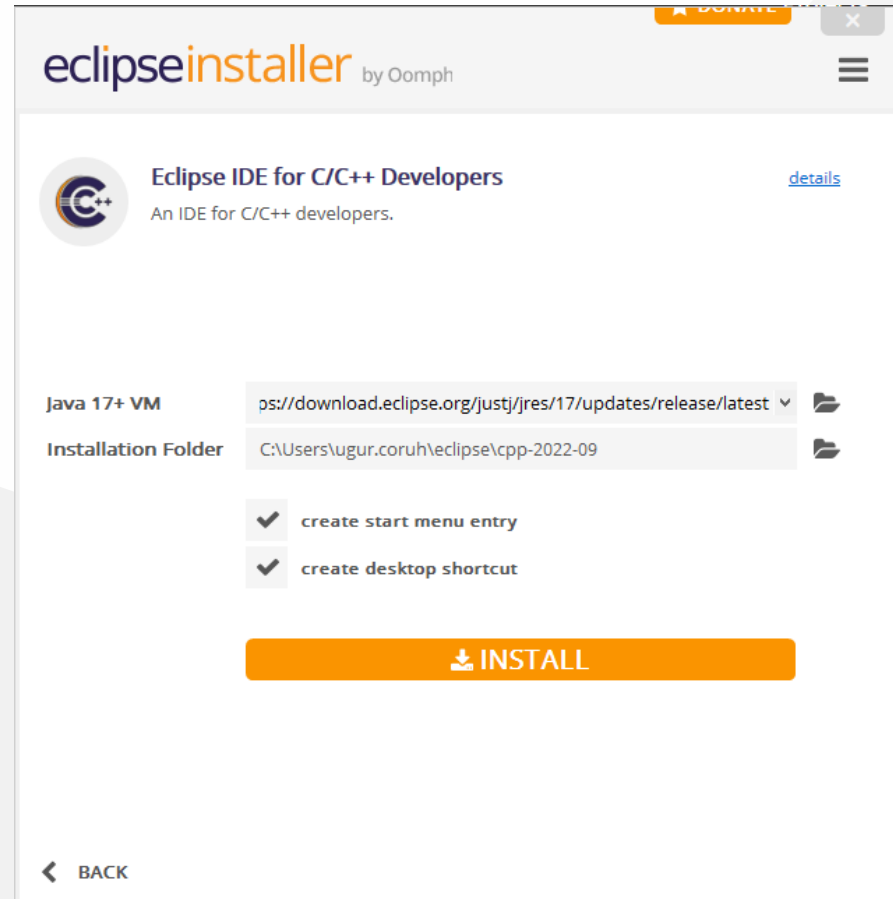
## Eclipse (C/C++) - Compile Only / Debugging Has Problem (1)

- Download and install Eclipse IDE from the following link
  - [Eclipse IDE for C/C++ Developers | Eclipse Packages](#)
- Run Installer
- Select Eclipse IDE for C/C++ Developers



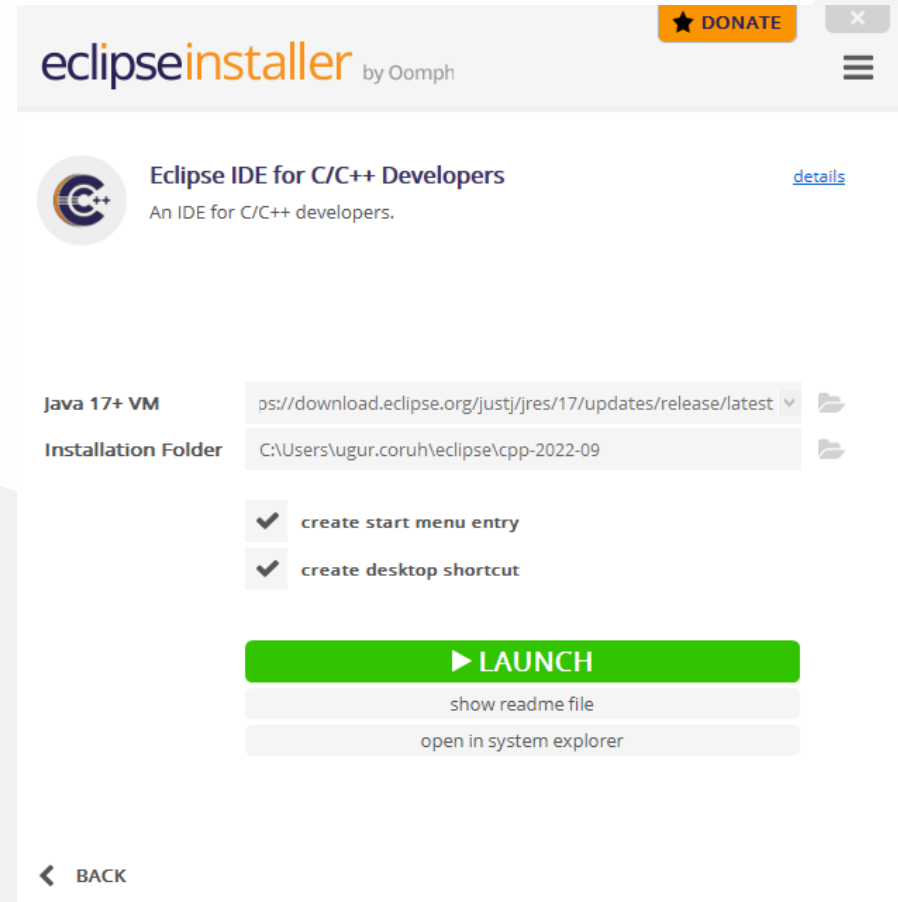
## Eclipse (C/C++) - Compile Only / Debugging Has Problem (2)

- Select Java Version and Installation Path



## Eclipse (C/C++) - Compile Only / Debugging Has Problem (3)

- After installation you can LAUNCH eclipse IDE

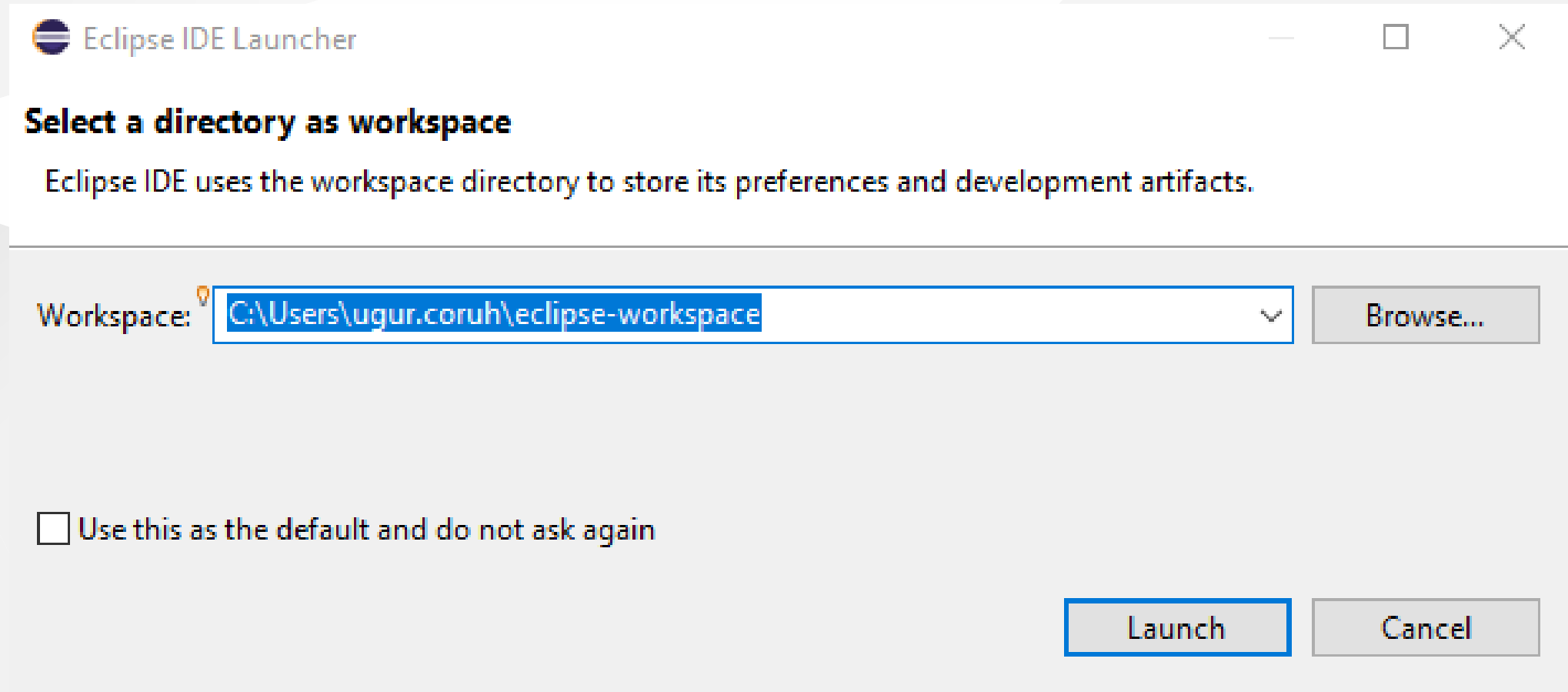


## Eclipse (C/C++) - Compile Only / Debugging Has Problem (4)



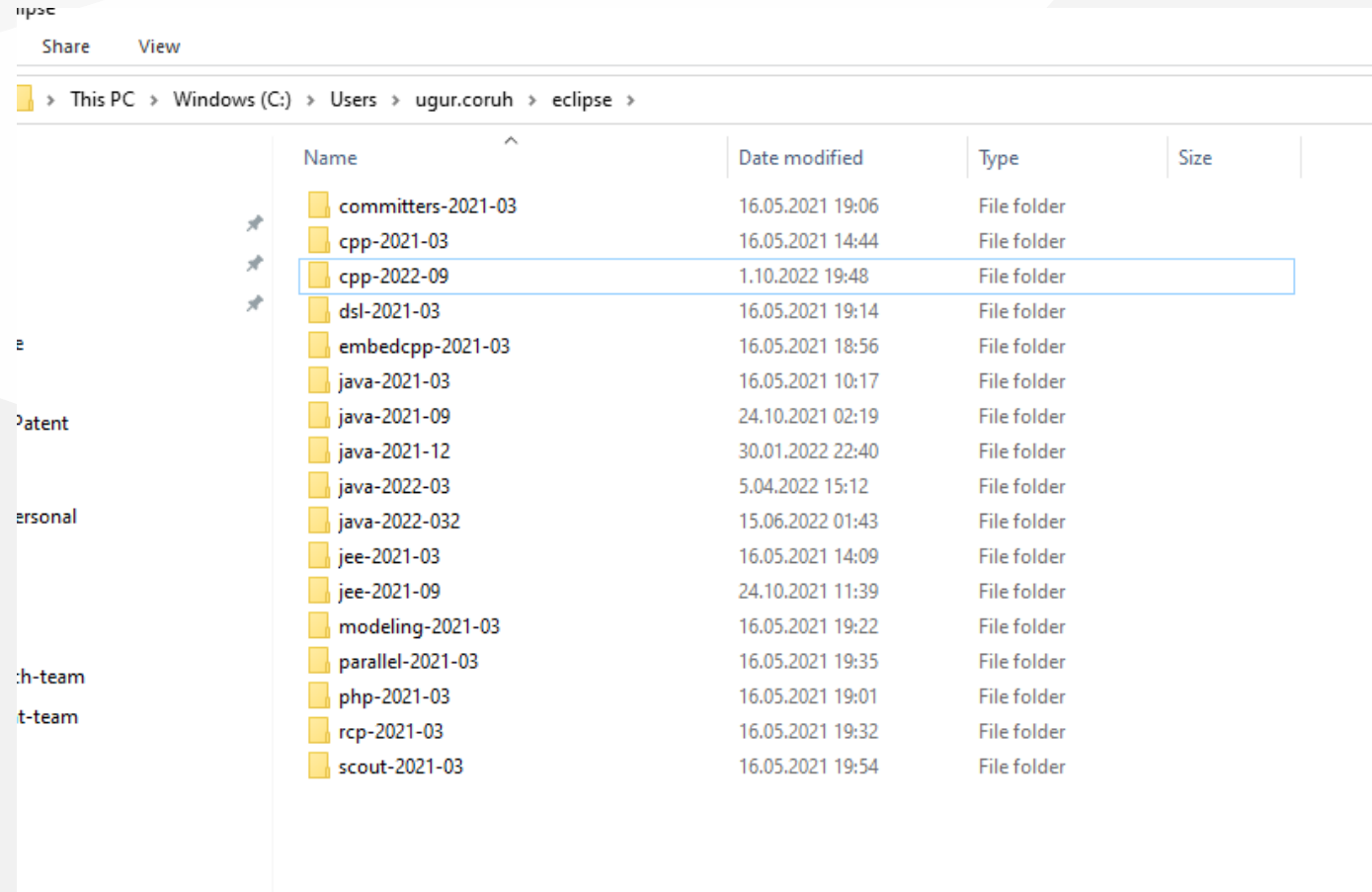
## Eclipse (C/C++) - Compile Only / Debugging Has Problem (5)

- Select a workspace that your project will be saved



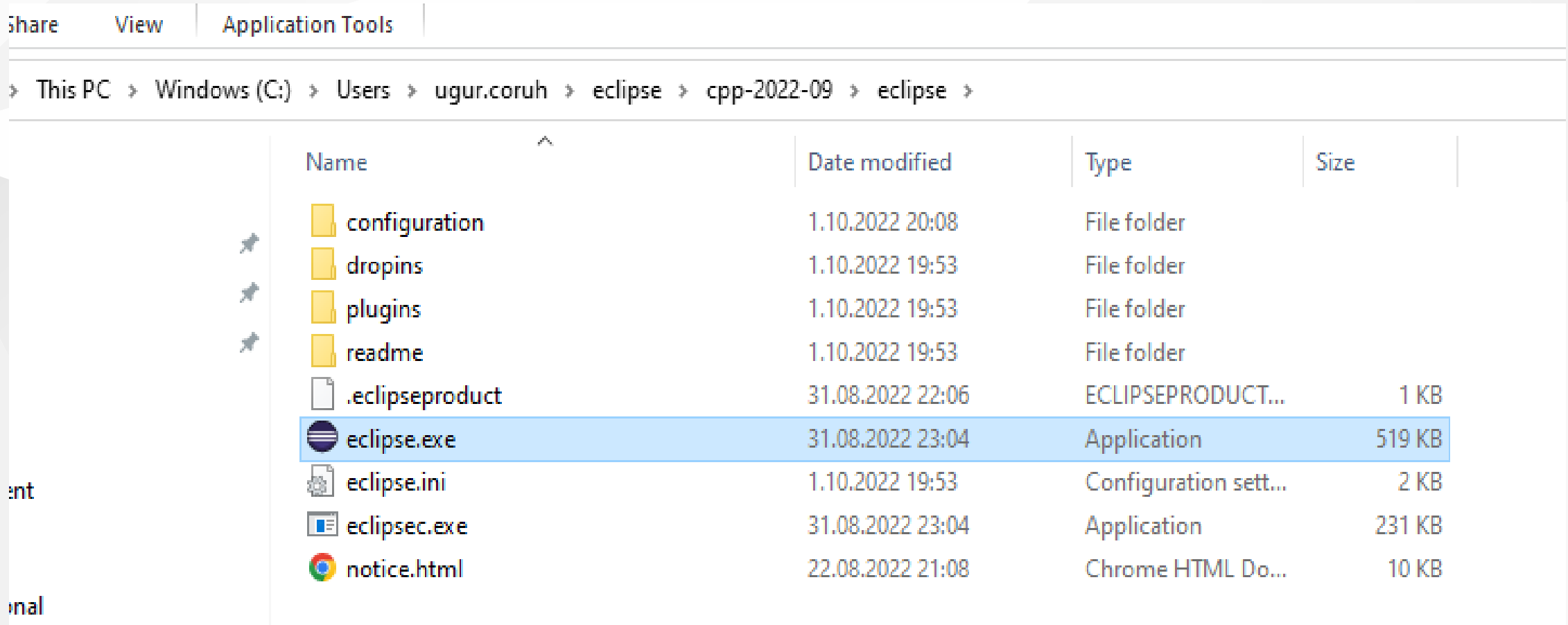
## Eclipse (C/C++) - Compile Only / Debugging Has Problem (6)

- You can find your installation under your user folder



## Eclipse (C/C++) - Compile Only / Debugging Has Problem (7)

- You can create shortcut to desktop for your working eclipse version.

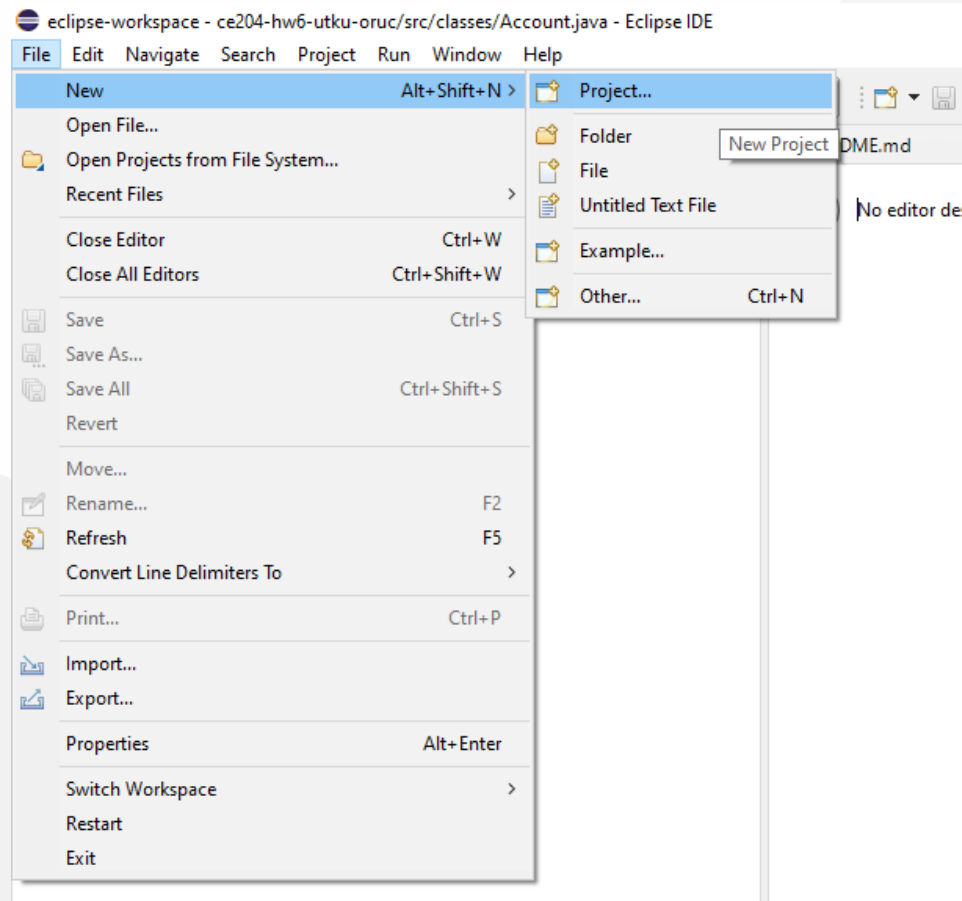


The screenshot shows a Windows File Explorer window with the following path: This PC > Windows (C:) > Users > ugur.coruh > eclipse > cpp-2022-09 > eclipse >. The window displays a list of files and folders. The 'eclipse.exe' file is highlighted in blue.

Name	Date modified	Type	Size
configuration	1.10.2022 20:08	File folder	
dropins	1.10.2022 19:53	File folder	
plugins	1.10.2022 19:53	File folder	
readme	1.10.2022 19:53	File folder	
.eclipseproduct	31.08.2022 22:06	ECLIPSEPRODUCT...	1 KB
<b>eclipse.exe</b>	31.08.2022 23:04	Application	519 KB
eclipse.ini	1.10.2022 19:53	Configuration sett...	2 KB
eclipsec.exe	31.08.2022 23:04	Application	231 KB
notice.html	22.08.2022 21:08	Chrome HTML Do...	10 KB

## Eclipse (C/C++) - Compile Only / Debugging Has Problem (8)

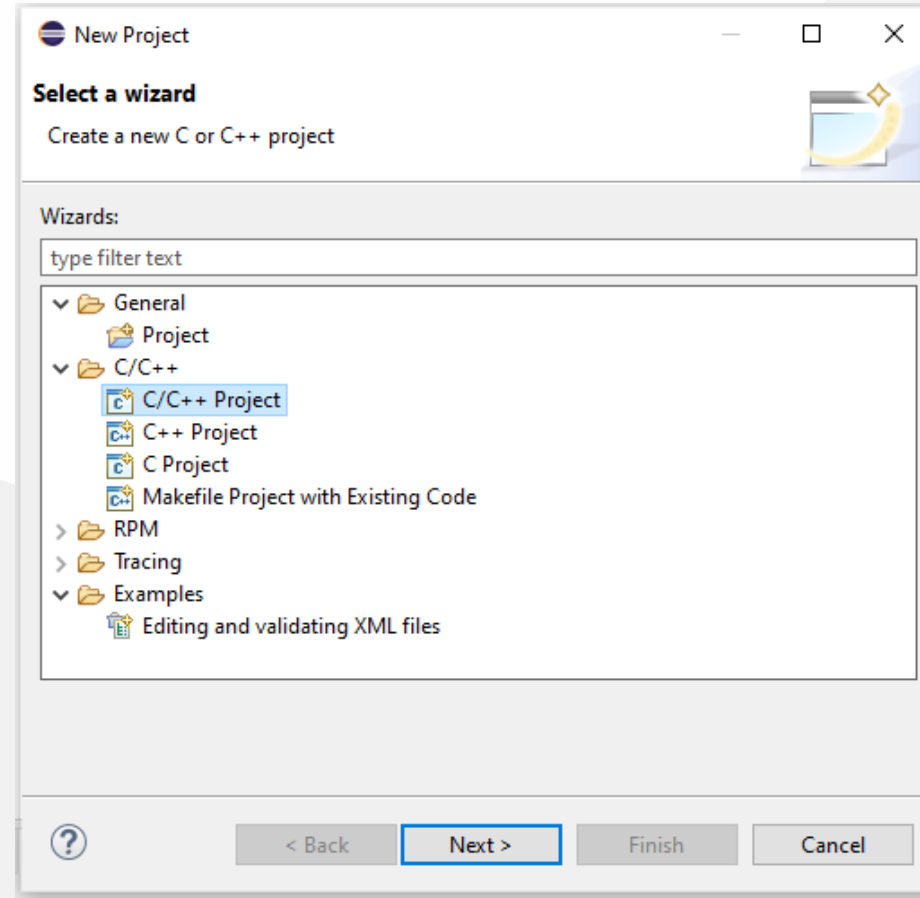
- File -> New -> Project





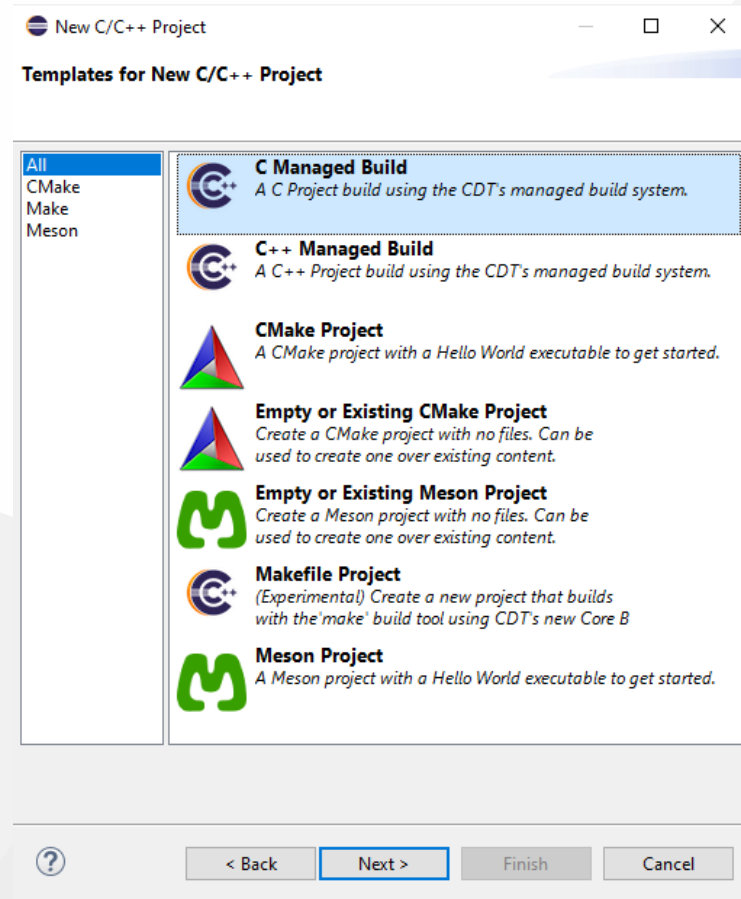
## Eclipse (C/C++) - Compile Only / Debugging Has Problem (9)

- Select C/C++ Project



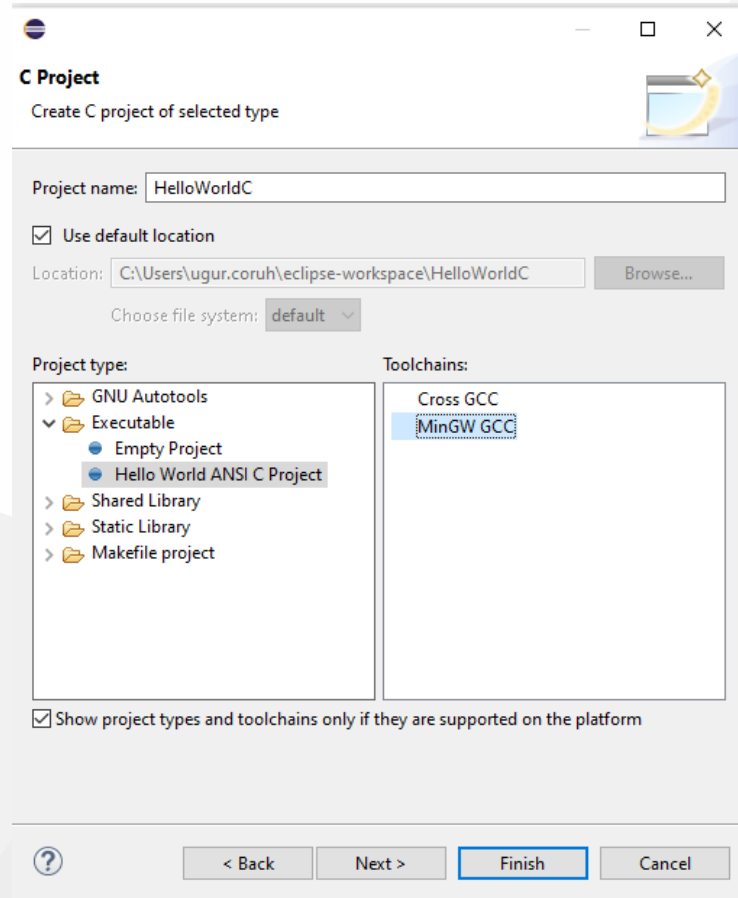
## Eclipse (C/C++) - Compile Only / Debugging Has Problem (10)

- Select C Managed Build, Eclipse CDT will do job for us.



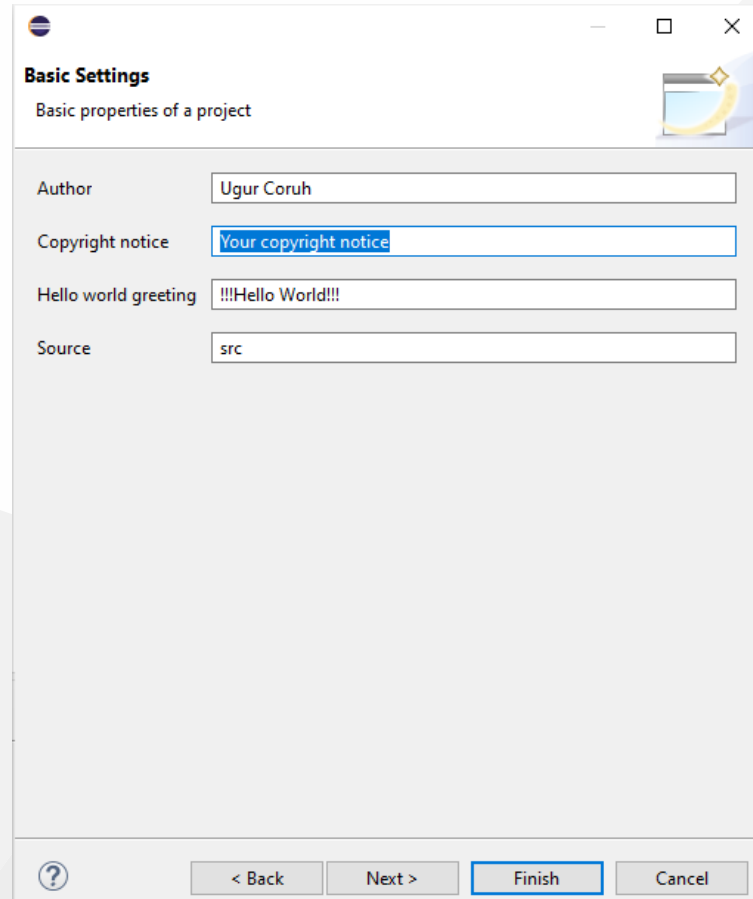
## Eclipse (C/C++) - Compile Only / Debugging Has Problem (11)

- Give project name and select a basic template executable with MinGW GCC.



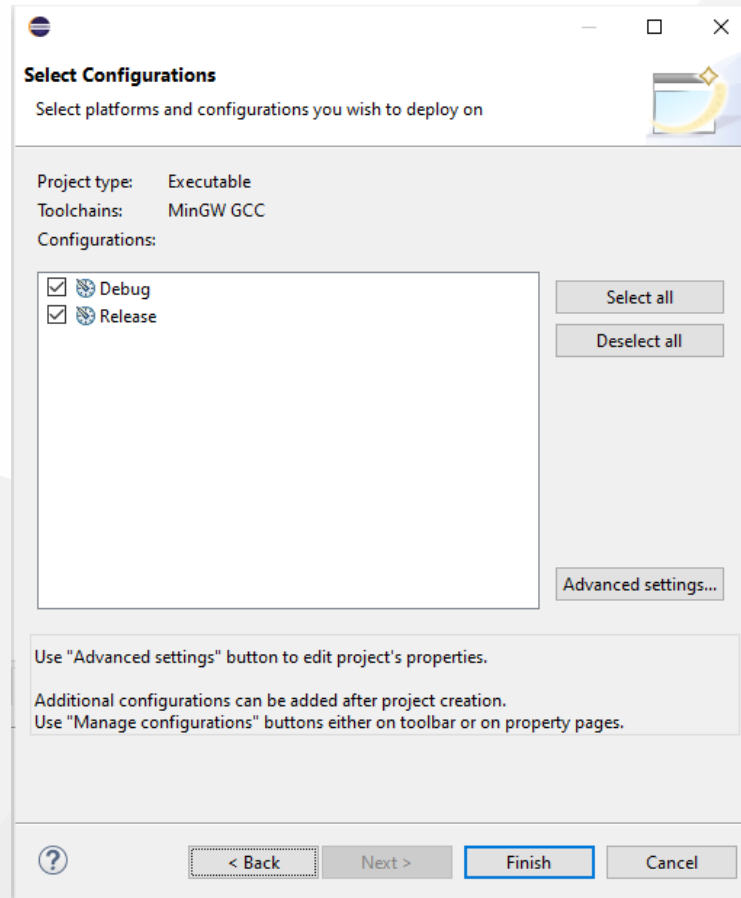
## Eclipse (C/C++) - Compile Only / Debugging Has Problem (12)

- Configura Basic Settings



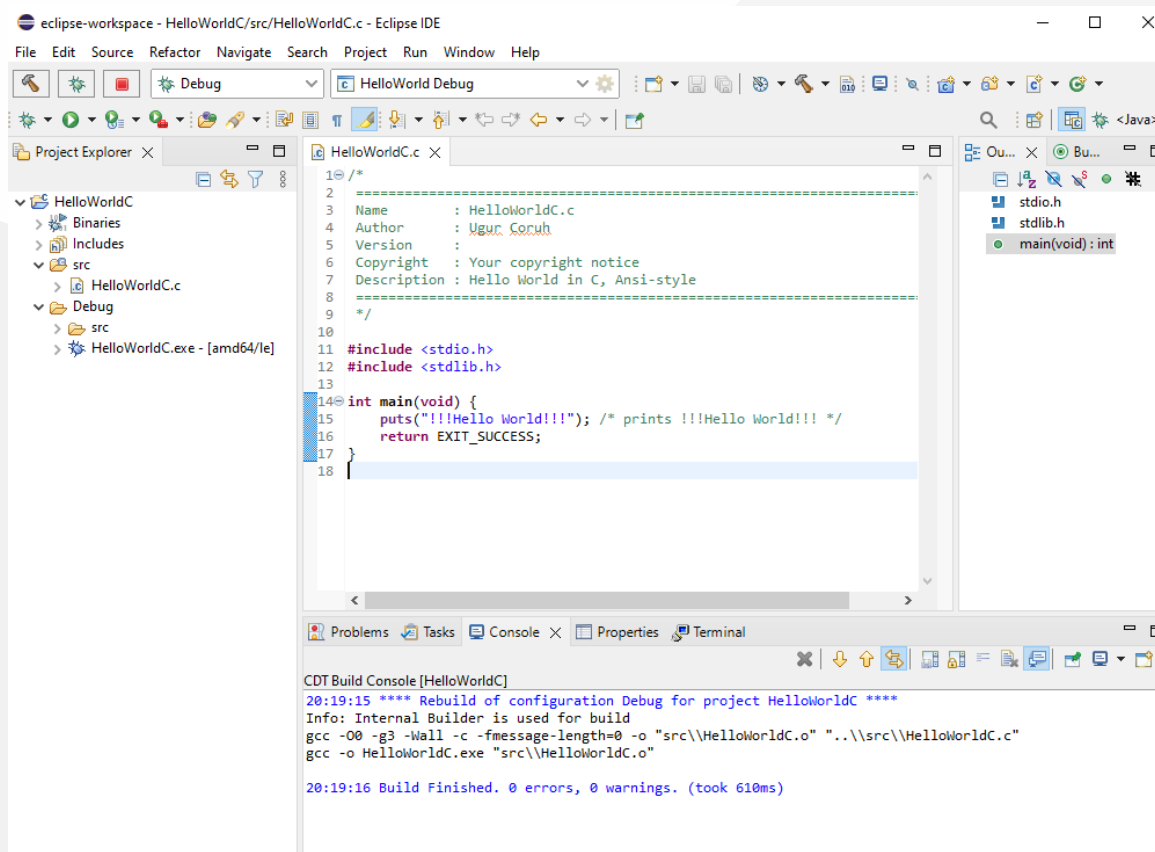
## Eclipse (C/C++) - Compile Only / Debugging Has Problem (13)

- There are default Debug and Release configurations you can add your customized configurations from Advanced Settings.



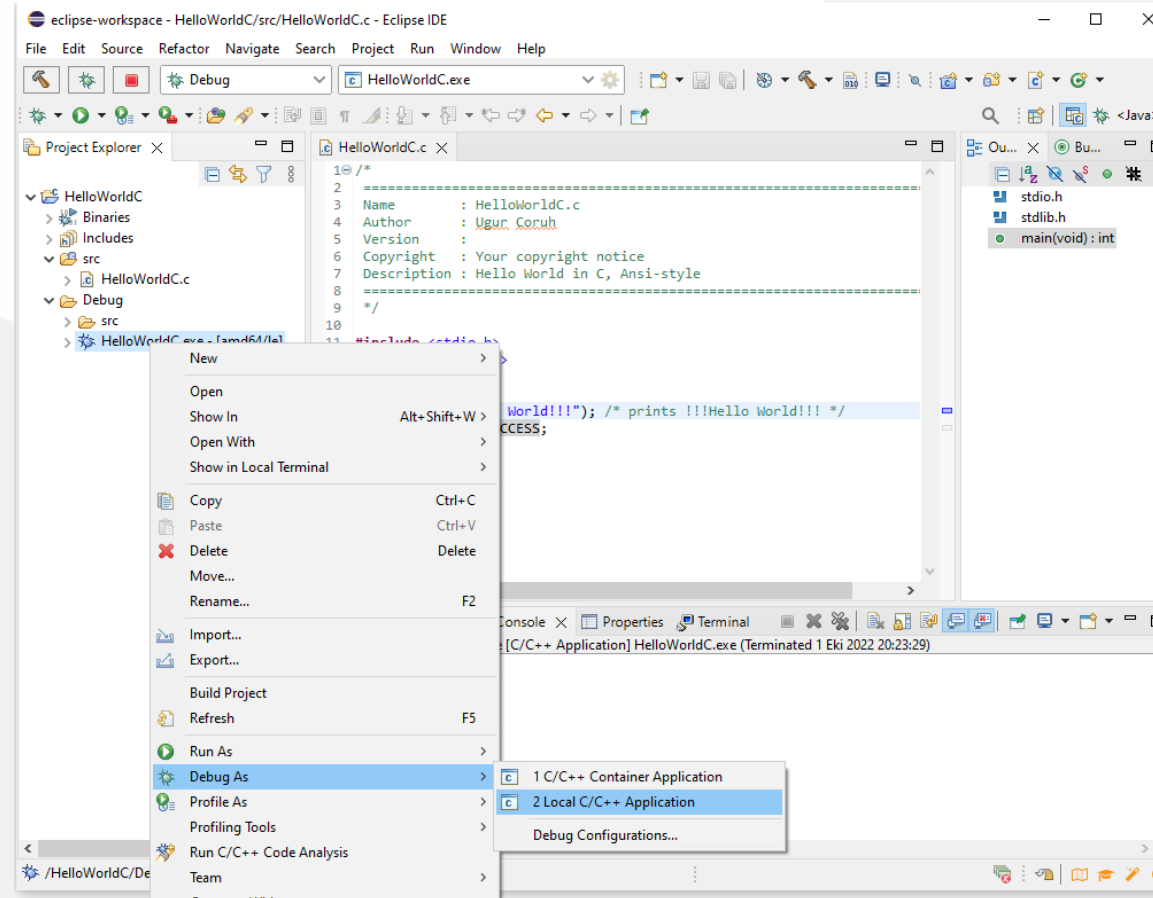
- Project settings will be C Select Debug/Release configuration and then Build Application Project->Build All (Ctrl+B)
- HelloWorldC.exe will be generated

```
gcc -O0 -g3 -Wall -c -fmessage-length=0 -o "src\\HelloWorldC.o" "..\\src\\HelloWorldC.c"  
gcc -o HelloWorldC.exe "src\\HelloWorldC.o"
```



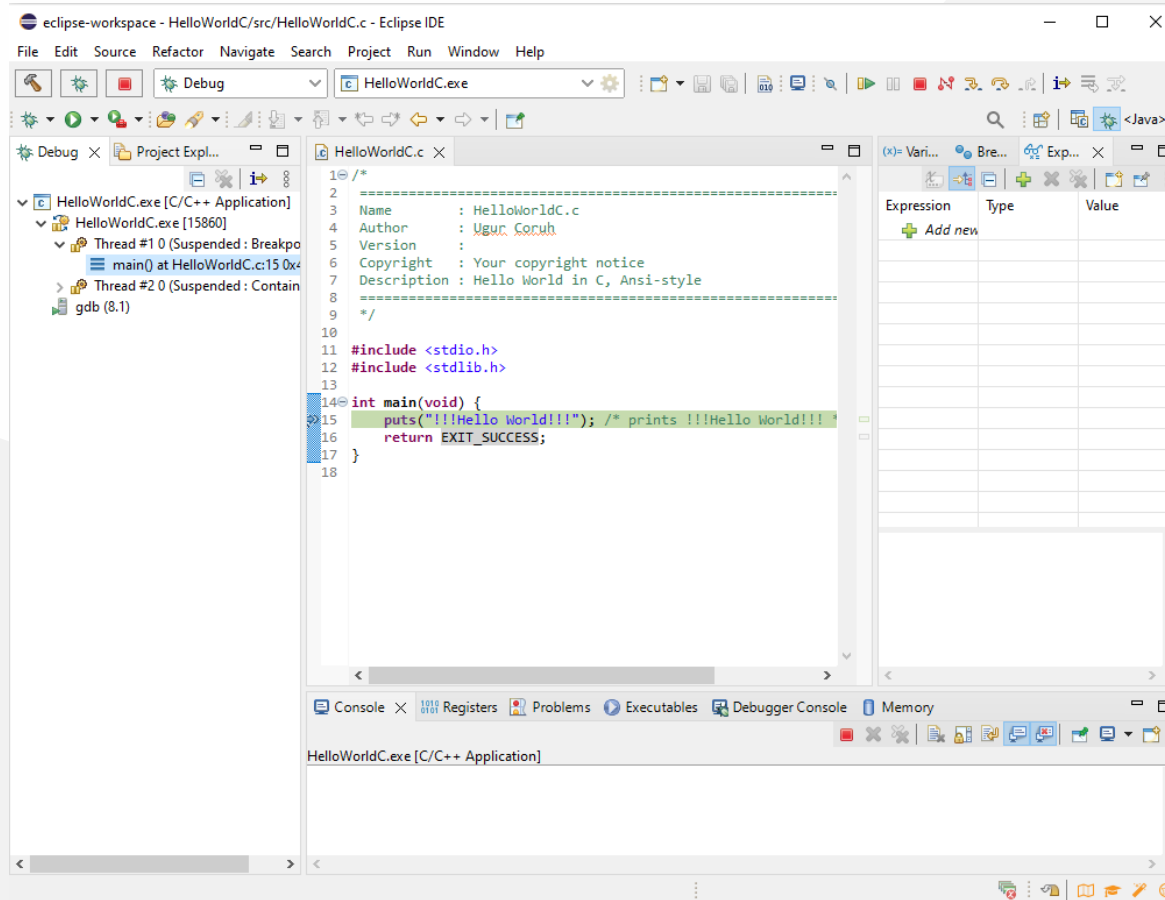
## CE103 Algorithms and Programming I Eclipse (C/C++) - Compile Only / Debugging Has Problem (15)

- Before build if you want to debug application select debug configuration, put your breakpoints and then Build application again.
- Right click the generated executable Debug As -> Local C/C++ Application



## Eclipse (C/C++) - Compile Only / Debugging Has Problem (16)

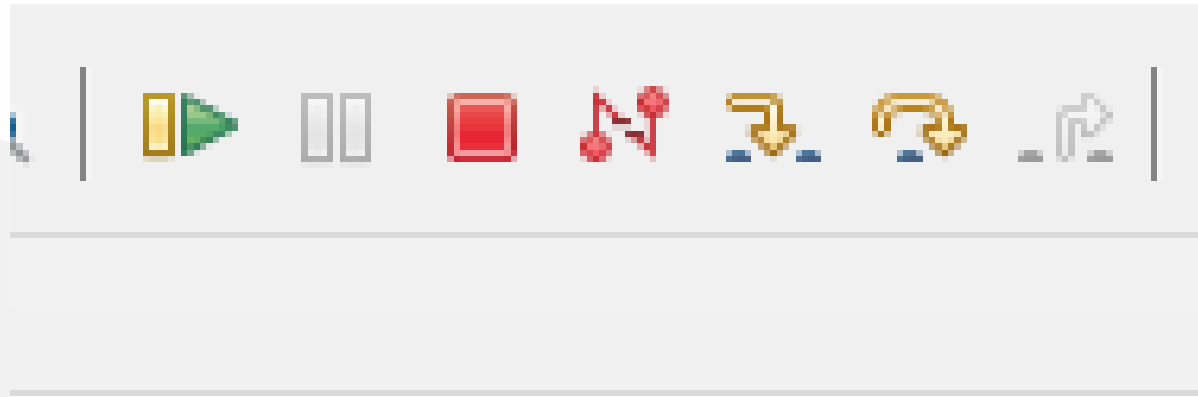
- Debugger will start and stop at breakpoint as follow.





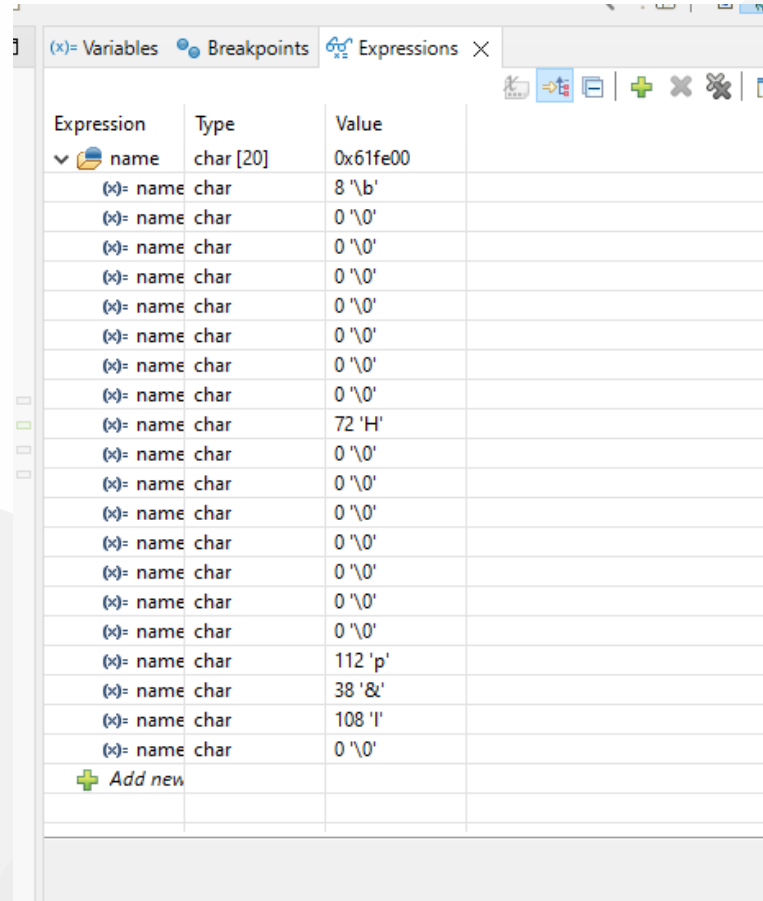
## Eclipse (C/C++) - Compile Only / Debugging Has Problem (16)

- Check debug control shortcuts and use them



## Eclipse (C/C++) - Compile Only / Debugging Has Problem (17)

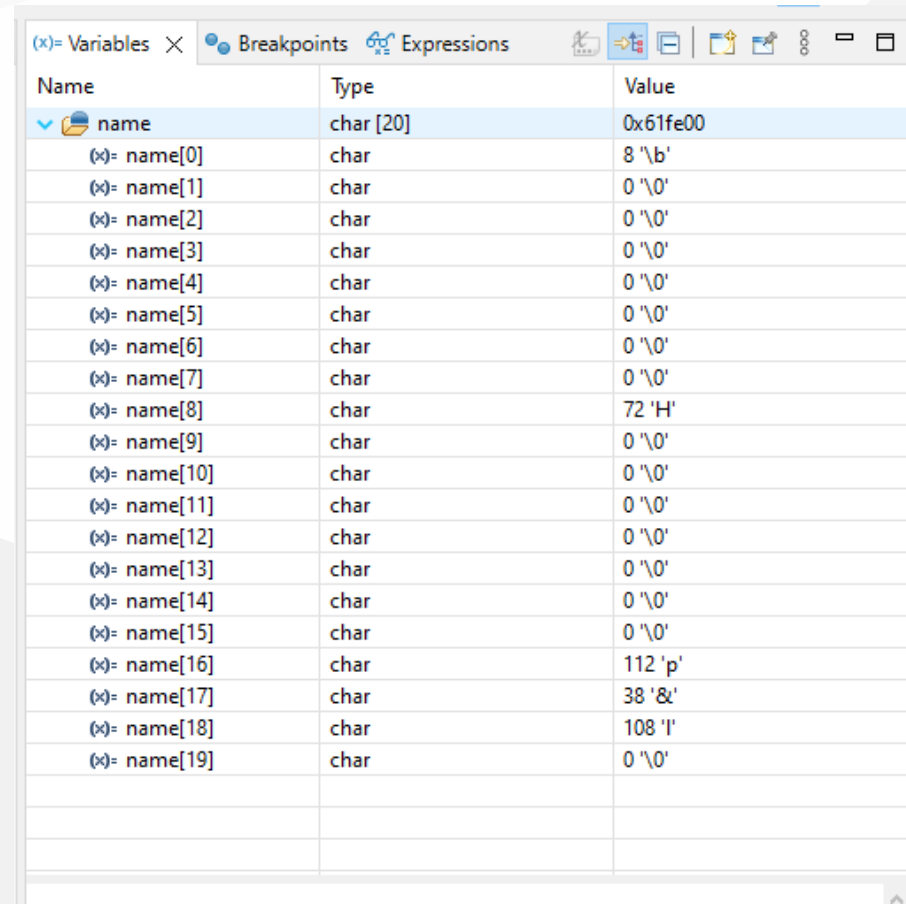
- To watch variables use Expressions and Variables



The screenshot shows the Eclipse IDE's Expressions view. The title bar indicates '(x)= Variables Breakpoints Expressions X'. The view contains a table with three columns: Expression, Type, and Value. The first row shows a variable 'name' of type 'char [20]' with a value of '0x61fe00'. Below this, the individual characters of the array are listed, each with a type of 'char' and a value. The values are: 8 '\b', 0 '\0', 0 '\0', 0 '\0', 0 '\0', 0 '\0', 0 '\0', 0 '\0', 72 'H', 0 '\0', 0 '\0', 0 '\0', 0 '\0', 0 '\0', 0 '\0', 0 '\0', 112 'p', 38 '&', 108 'l', and 0 '\0'. At the bottom of the table, there is a '+ Add new' button.

Expression	Type	Value
name	char [20]	0x61fe00
(x)= name	char	8 '\b'
(x)= name	char	0 '\0'
(x)= name	char	0 '\0'
(x)= name	char	0 '\0'
(x)= name	char	0 '\0'
(x)= name	char	0 '\0'
(x)= name	char	0 '\0'
(x)= name	char	0 '\0'
(x)= name	char	72 'H'
(x)= name	char	0 '\0'
(x)= name	char	0 '\0'
(x)= name	char	0 '\0'
(x)= name	char	0 '\0'
(x)= name	char	0 '\0'
(x)= name	char	0 '\0'
(x)= name	char	112 'p'
(x)= name	char	38 '&'
(x)= name	char	108 'l'
(x)= name	char	0 '\0'
+ Add new		

## Eclipse (C/C++) - Compile Only / Debugging Has Problem (18)



The screenshot shows the Eclipse IDE's Variables view. The variable 'name' is expanded, showing its elements. The array contains the string 'bH&I' followed by null characters.

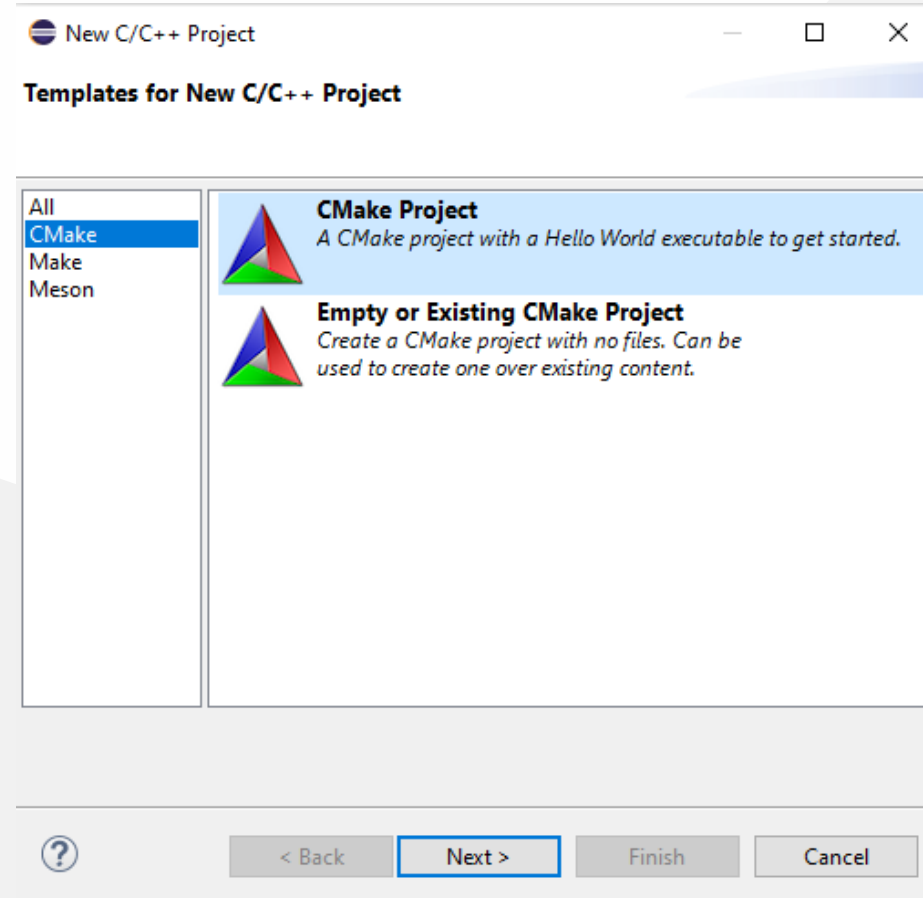
Name	Type	Value
name	char [20]	0x61fe00
(x)- name[0]	char	8 'b'
(x)- name[1]	char	0 '\0'
(x)- name[2]	char	0 '\0'
(x)- name[3]	char	0 '\0'
(x)- name[4]	char	0 '\0'
(x)- name[5]	char	0 '\0'
(x)- name[6]	char	0 '\0'
(x)- name[7]	char	0 '\0'
(x)- name[8]	char	72 'H'
(x)- name[9]	char	0 '\0'
(x)- name[10]	char	0 '\0'
(x)- name[11]	char	0 '\0'
(x)- name[12]	char	0 '\0'
(x)- name[13]	char	0 '\0'
(x)- name[14]	char	0 '\0'
(x)- name[15]	char	0 '\0'
(x)- name[16]	char	112 'p'
(x)- name[17]	char	38 '&'
(x)- name[18]	char	108 'I'
(x)- name[19]	char	0 '\0'

## Eclipse (C/C++) - Compile Only / Debugging Has Problem (19)

- for more visit eclipse webpage
  - [Effective Techniques for Debugging C & C++ | The Eclipse Foundation](#)
  - [Help - Eclipse IDE](#)

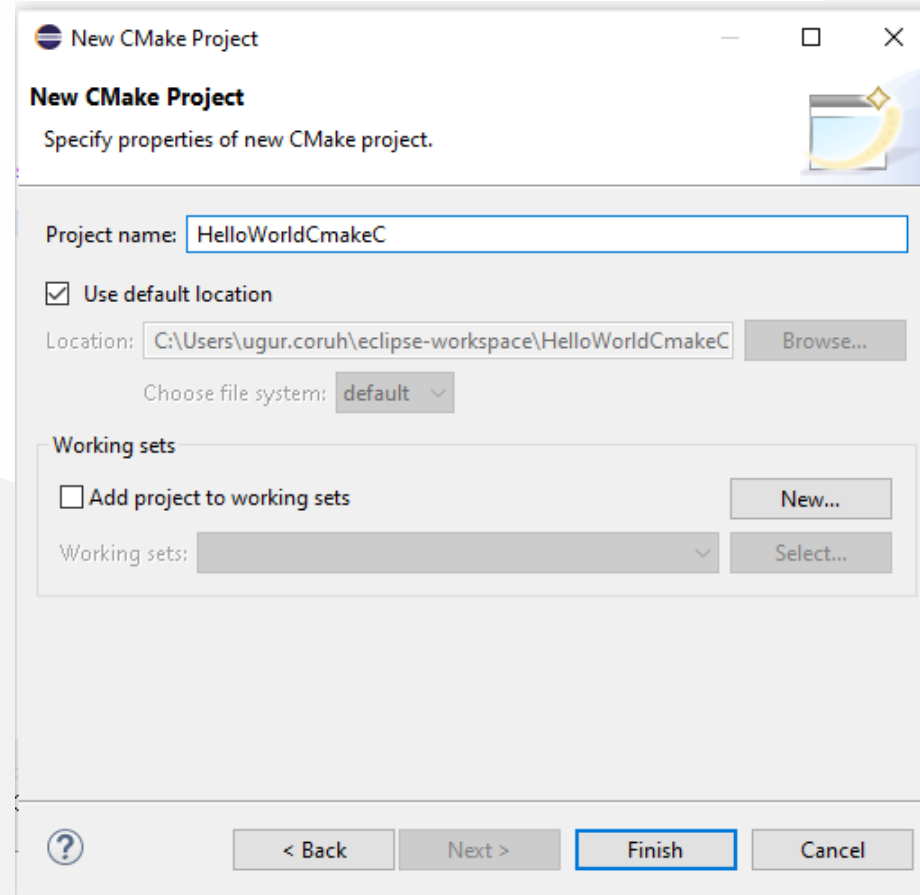
## Eclipse (C/C++) - Compile Only / Debugging Has Problem (20)

- Generate CMAKE project from new Project and Select CMake Project Template



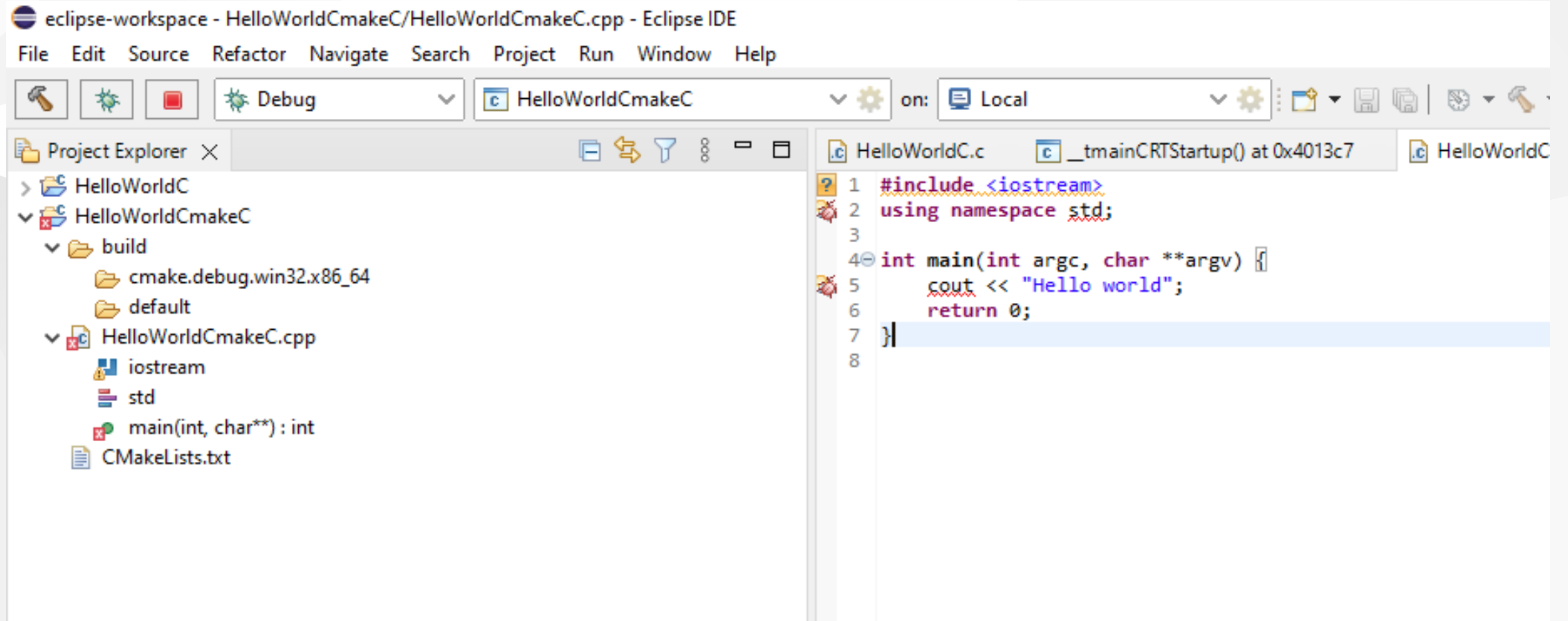
## Eclipse (C/C++) - Compile Only / Debugging Has Problem (21)

- Give project name



## Eclipse (C/C++) - Compile Only / Debugging Has Problem (22)

- This will generate default C++ Hello World project



eclipse-workspace - HelloWorldCmakeC/HelloWorldCmakeC.cpp - Eclipse IDE

File Edit Source Refactor Navigate Search Project Run Window Help

Debug HelloWorldCmakeC on: Local

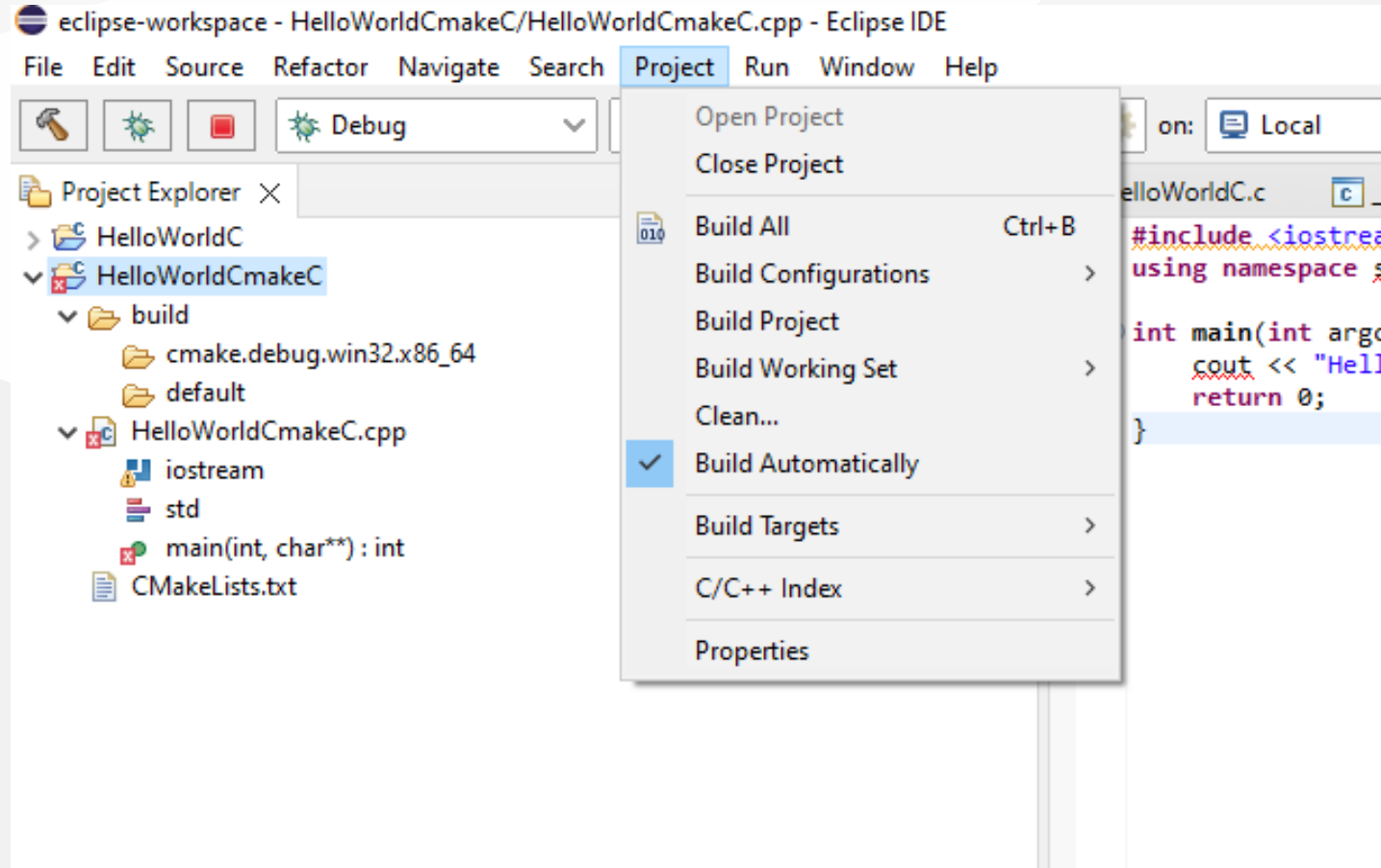
Project Explorer

- HelloWorldC
- HelloWorldCmakeC
  - build
    - cmake.debug.win32.x86\_64
    - default
  - HelloWorldCmakeC.cpp
    - iostream
    - std
    - main(int, char\*\*) : int
    - CMakeLists.txt

```
1 #include <iostream>
2 using namespace std;
3
4 int main(int argc, char **argv) {
5     cout << "Hello world";
6     return 0;
7 }
8
```

## Eclipse (C/C++) - Compile Only / Debugging Has Problem (23)

- Build Project





## Eclipse (C/C++) - Compile Only / Debugging Has Problem (24)

- It will give following errors, for missing configurations. This errors are generated by CMAKE
- Then clean and rebuild again.

```
Errors occurred during the build.
```

```
Errors running builder 'CDT Core Builder' on project 'HelloWorldCmakeC'.
```

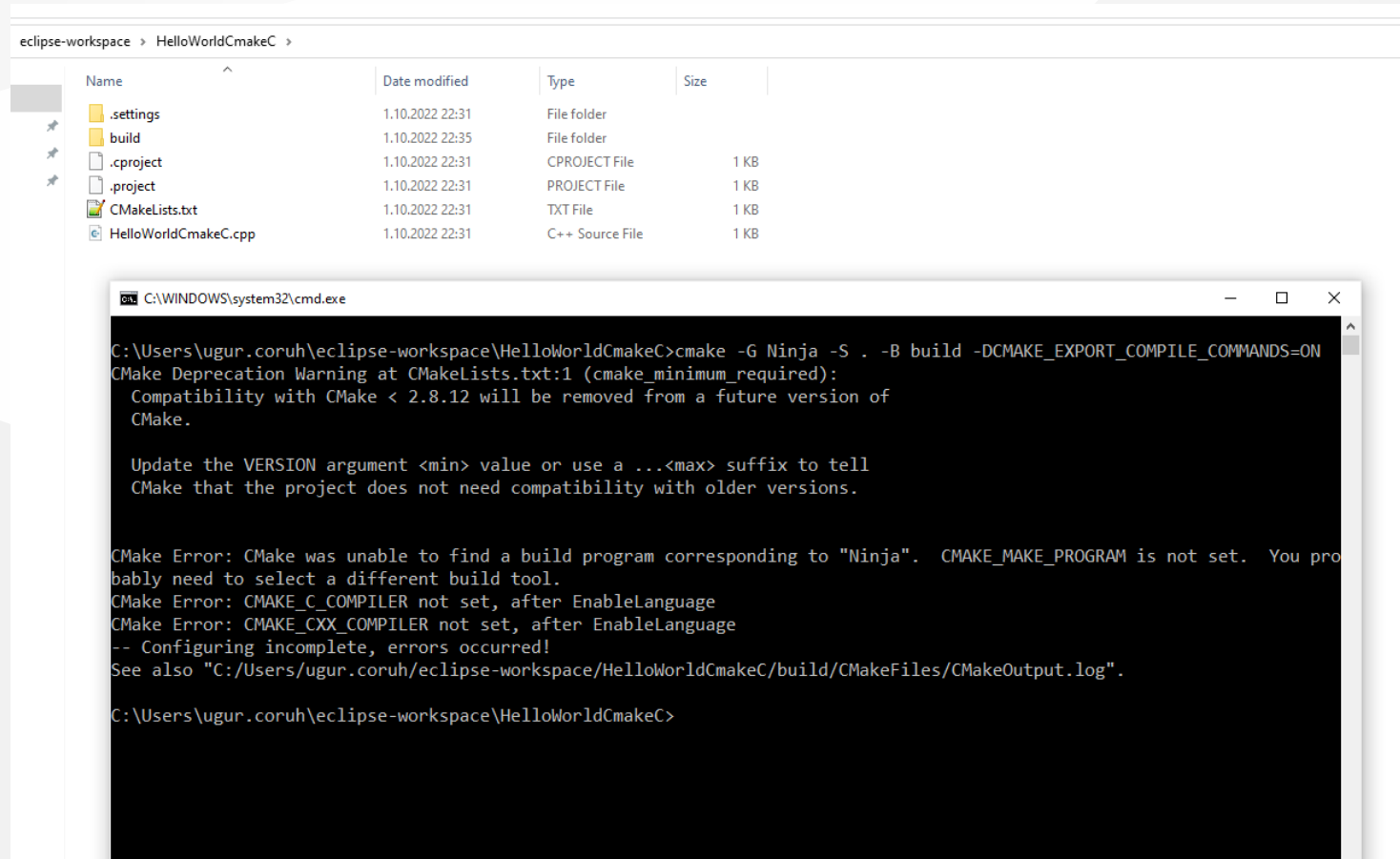
```
Resource '/HelloWorldCmakeC/build/cmake.debug.win32.x86_64/compile_commands.json' does not exist.
```

```
Resource '/HelloWorldCmakeC/build/cmake.debug.win32.x86_64/compile_commands.json' does not exist.
```

```
Resource '/HelloWorldCmakeC/build/cmake.debug.win32.x86_64/compile_commands.json' does not exist.
```

```
Resource '/HelloWorldCmakeC/build/cmake.debug.win32.x86_64/compile_commands.json' does not exist.
```

## Eclipse (C/C++) - Compile Only / Debugging Has Problem (25)



The screenshot shows the Eclipse IDE interface. The top part displays a file explorer for the workspace 'eclipse-workspace' > 'HelloWorldCmakeC'. The files listed are:

Name	Date modified	Type	Size
.settings	1.10.2022 22:31	File folder	
build	1.10.2022 22:35	File folder	
.cproject	1.10.2022 22:31	CPROJECT File	1 KB
.project	1.10.2022 22:31	PROJECT File	1 KB
CMakeLists.txt	1.10.2022 22:31	TXT File	1 KB
HelloWorldCmakeC.cpp	1.10.2022 22:31	C++ Source File	1 KB

Below the file explorer is a terminal window titled 'C:\WINDOWS\system32\cmd.exe'. It shows the execution of the CMake command:

```
C:\Users\ugur.coruh\eclipse-workspace\HelloWorldCmakeC>cmake -G Ninja -S . -B build -DCMAKE_EXPORT_COMPILE_COMMANDS=ON
CMake Deprecation Warning at CMakeLists.txt:1 (cmake_minimum_required):
  Compatibility with CMake < 2.8.12 will be removed from a future version of
  CMake.

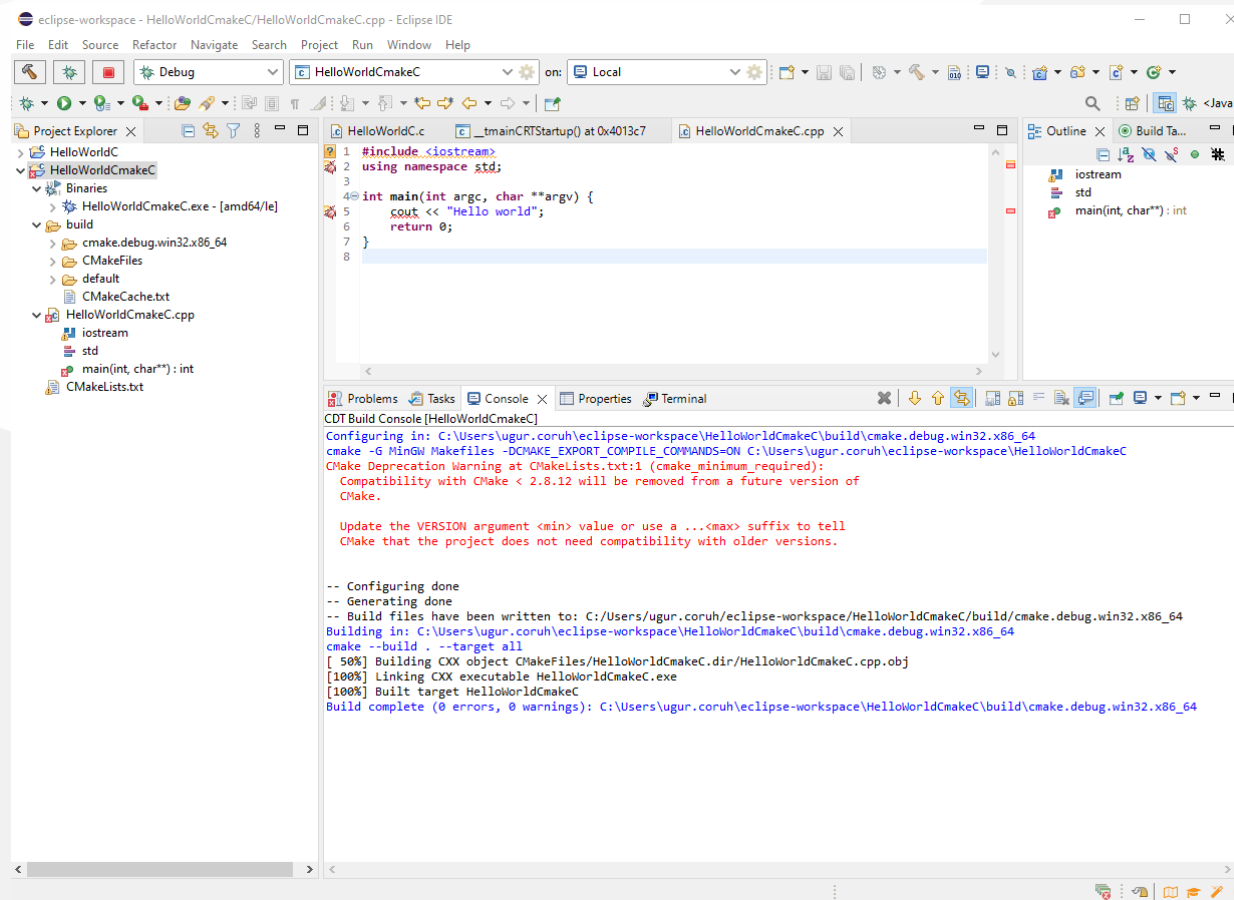
  Update the VERSION argument <min> value or use a ...<max> suffix to tell
  CMake that the project does not need compatibility with older versions.

CMake Error: CMake was unable to find a build program corresponding to "Ninja".  CMAKE_MAKE_PROGRAM is not set.  You probably
need to select a different build tool.
CMake Error: CMAKE_C_COMPILER not set, after EnableLanguage
CMake Error: CMAKE_CXX_COMPILER not set, after EnableLanguage
-- Configuring incomplete, errors occurred!
See also "C:/Users/ugur.coruh/eclipse-workspace/HelloWorldCmakeC/build/CMakeFiles/CMakeOutput.log".

C:\Users\ugur.coruh\eclipse-workspace\HelloWorldCmakeC>
```

## Eclipse (C/C++) - Compile Only / Debugging Has Problem (26)

- After this operation first Clean project from Project menu and then Build All again



```
eclipse-workspace - HelloWorldCmakeC/HelloWorldCmakeC.cpp - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Debug HelloWorldCmakeC on: Local
Project Explorer
  HelloWorldC
  HelloWorldCmakeC
    Binaries
      HelloWorldCmakeC.exe - [amd64/le]
    build
      cmake.debug.win32.x86_64
      CMakeFiles
      default
      CMakeCache.txt
    HelloWorldCmakeC.cpp
      iostream
      std
      main(int, char**): int
      CMakeLists.txt
HelloWorldC.cpp
  1 #include <iostream>
  2 using namespace std;
  3
  4 int main(int argc, char **argv) {
  5     cout << "Hello world";
  6     return 0;
  7 }
  8
CDT Build Console [HelloWorldCmakeC]
Configuring in: C:\Users\ugur.coruh\eclipse-workspace\HelloWorldCmakeC\build\cmake.debug.win32.x86_64
cmake -G MinGW Makefiles -DCMAKE_EXPORT_COMPILE_COMMANDS=ON C:\Users\ugur.coruh\eclipse-workspace\HelloWorldCmakeC
CMake Deprecation Warning at CMakeLists.txt:1 (cmake_minimum_required):
Compatibility with CMake < 2.8.12 will be removed from a future version of
CMake.

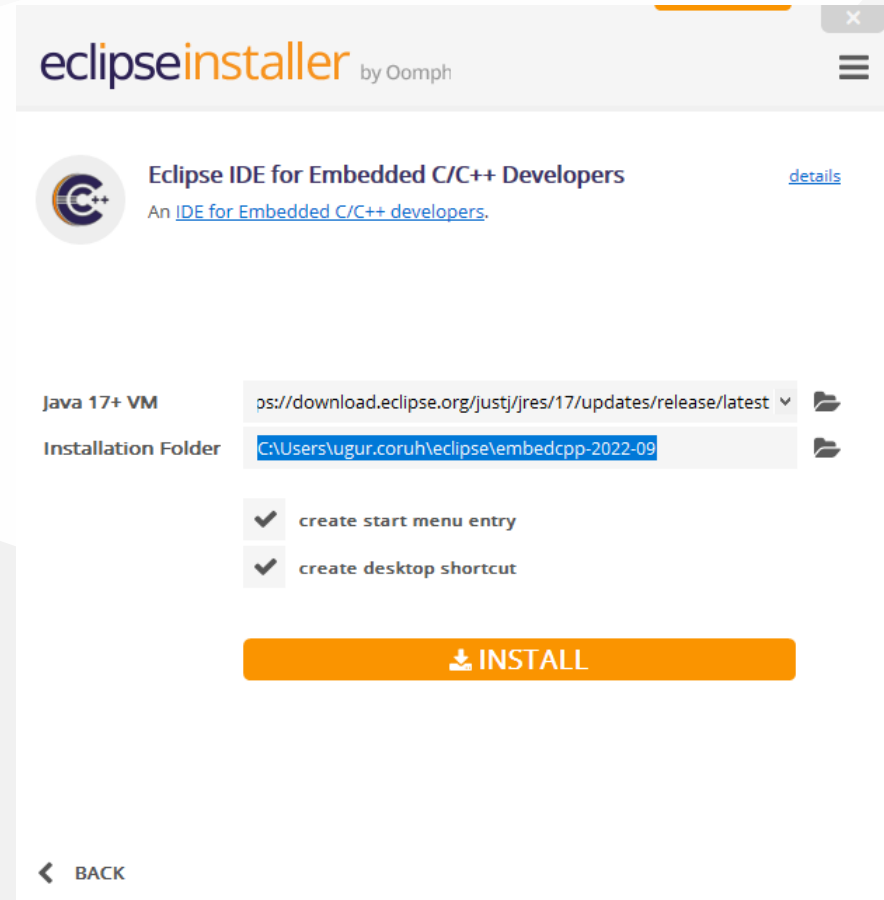
Update the VERSION argument <min> value or use a ...<max> suffix to tell
CMake that the project does not need compatibility with older versions.

-- Configuring done
-- Generating done
-- Build files have been written to: C:\Users\ugur.coruh\eclipse-workspace\HelloWorldCmakeC\build\cmake.debug.win32.x86_64
Building in: C:\Users\ugur.coruh\eclipse-workspace\HelloWorldCmakeC\build\cmake.debug.win32.x86_64
cmake --build . --target all
[ 50%] Building CXX object CMakeFiles/HelloWorldCmakeC.dir/HelloWorldCmakeC.cpp.obj
[100%] Linking CXX executable HelloWorldCmakeC.exe
[100%] Built target HelloWorldCmakeC
Build complete (0 errors, 0 warnings): C:\Users\ugur.coruh\eclipse-workspace\HelloWorldCmakeC\build\cmake.debug.win32.x86_64
```

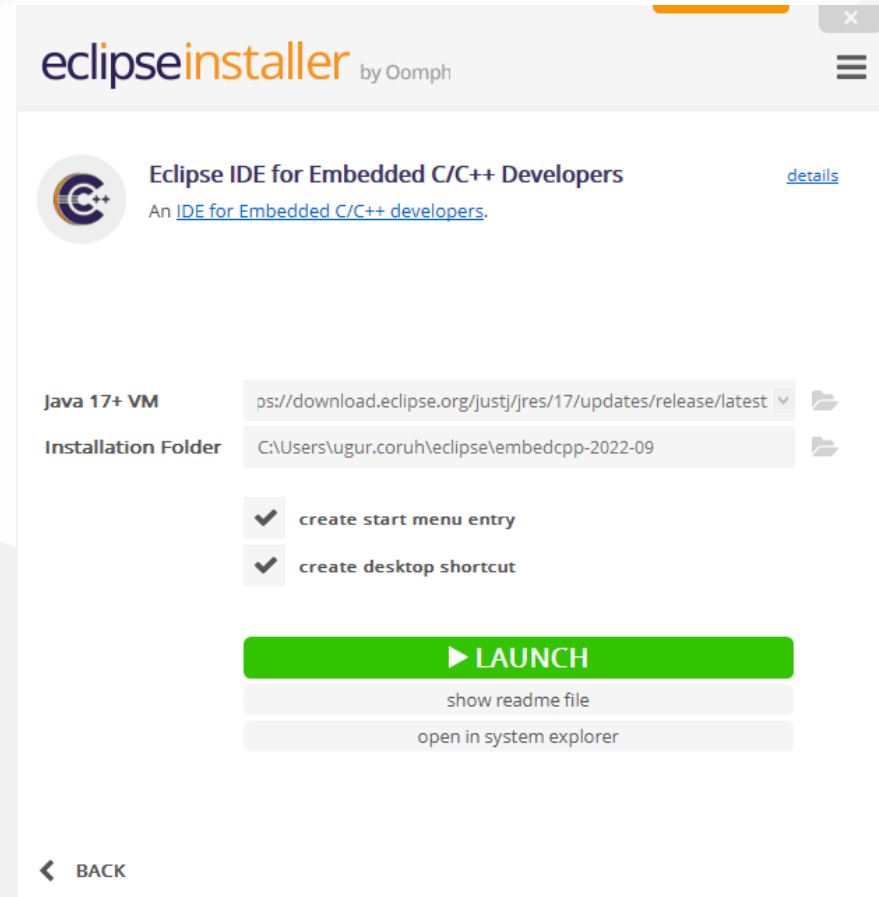
## Eclipse (C/C++) - Compile Only / Debugging Has Problem (27)

- Eclipse with CMake project on windows
- JV - Science and stuff.

## Eclipse (C/C++) - Compile Only / Debugging Has Problem (28)

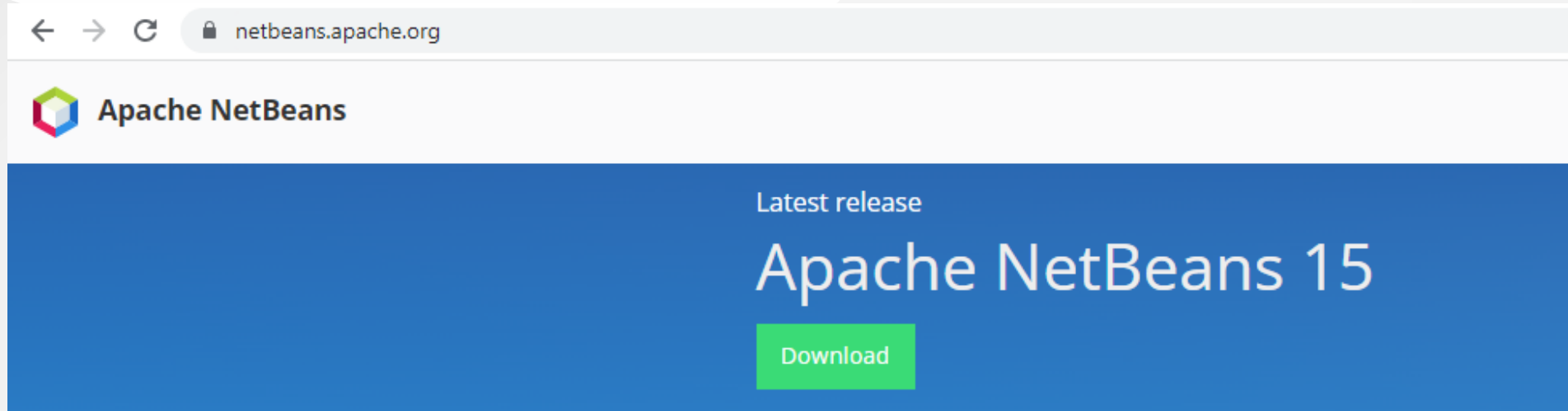


## Eclipse (C/C++) - Compile Only / Debugging Has Problem (29)

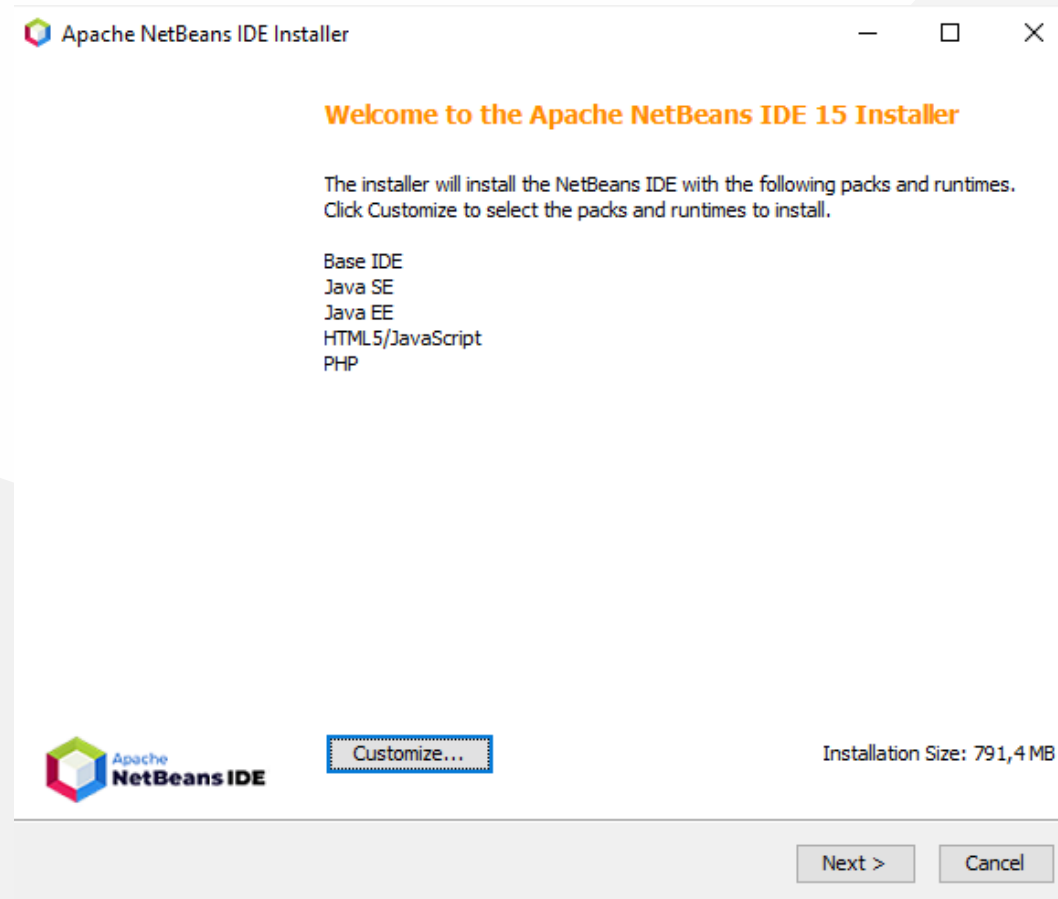


## Netbeans (C/C++) - Manuel Build/Clean/Run Command Setting Not Good Option for C/C++ Development (1)

- <https://netbeans.apache.org/>
- C and C++ Tutorials

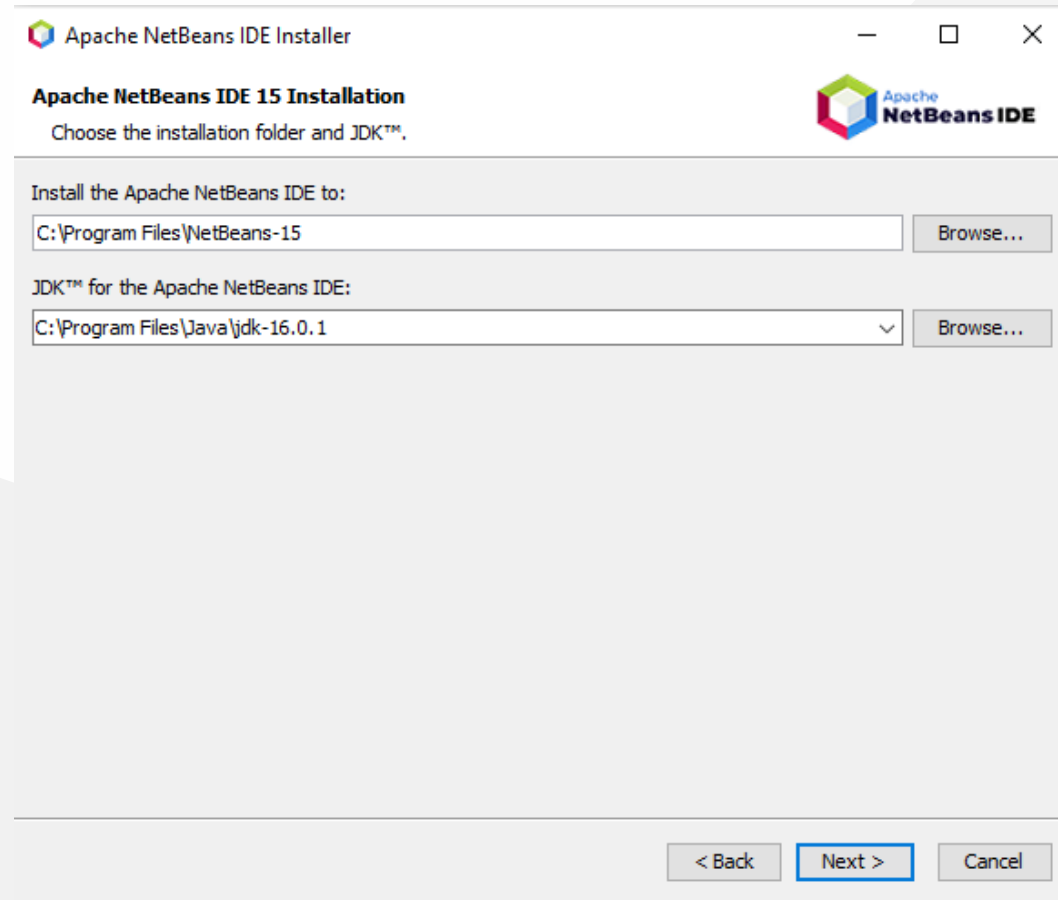


## Netbeans (C/C++) - Manuel Build/Clean/Run Command Setting Not Good Option for C/C++ Development (2)

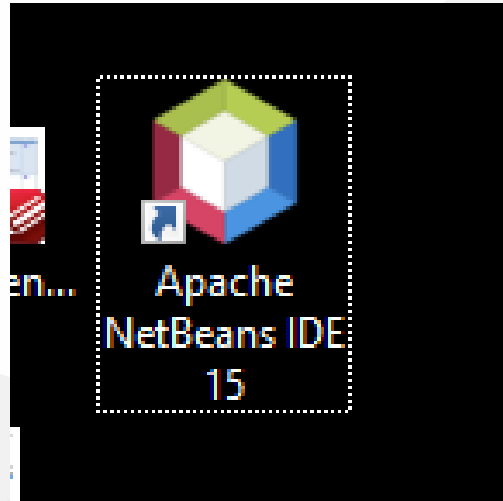




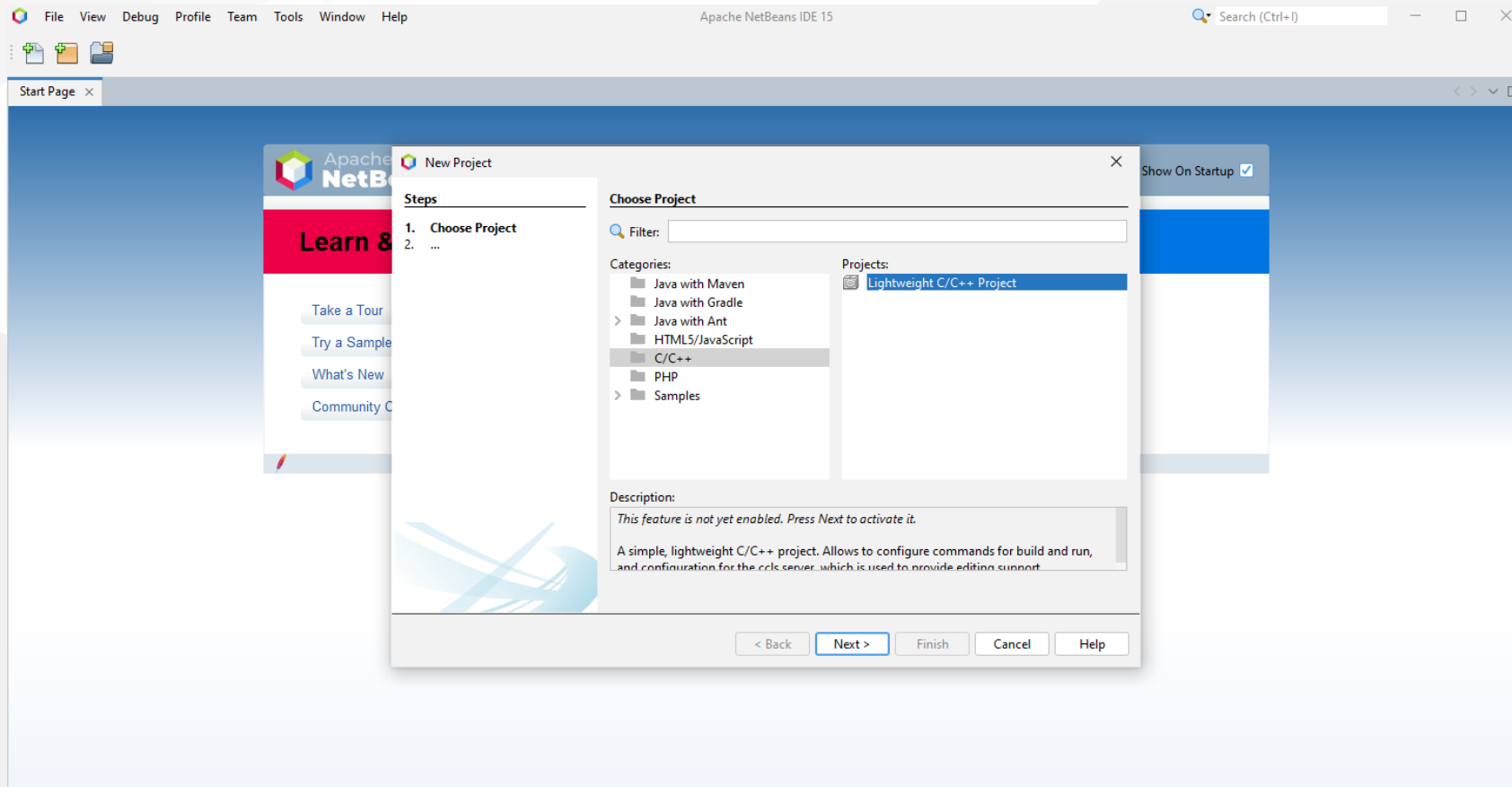
## Netbeans (C/C++) - Manuel Build/Clean/Run Command Setting Not Good Option for C/C++ Development (3)



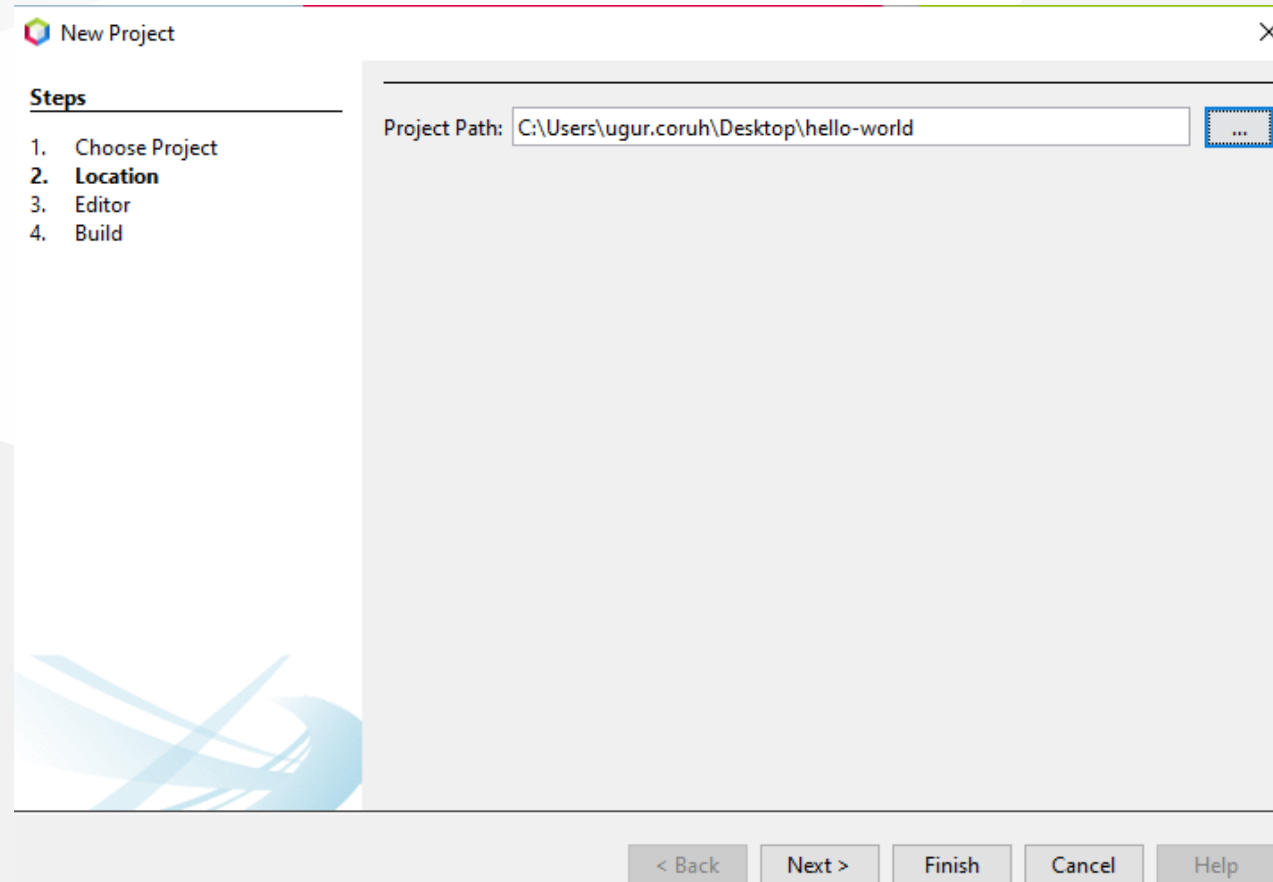
## Netbeans (C/C++) - Manuel Build/Clean/Run Command Setting Not Good Option for C/C++ Development (4)



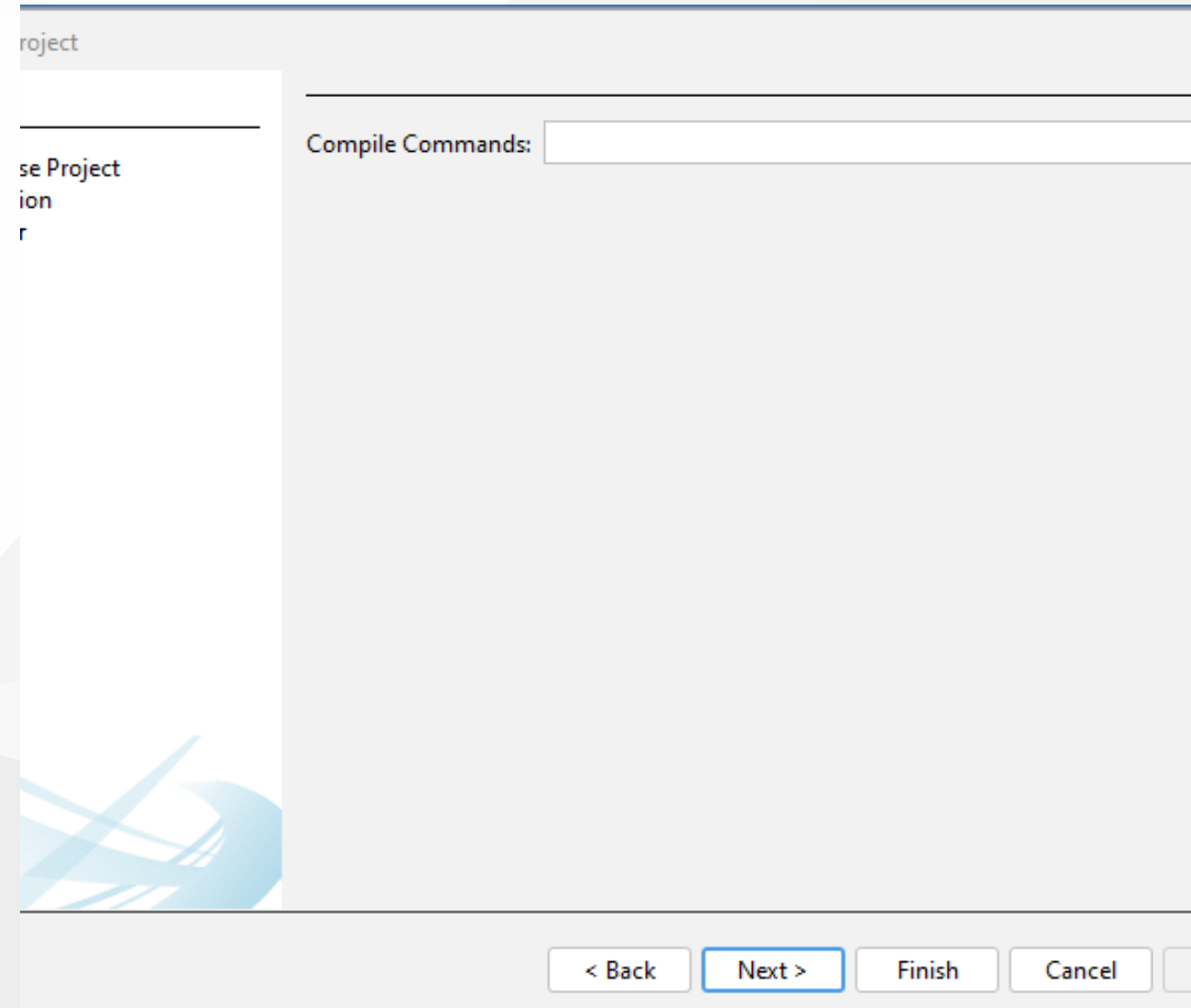
# Netbeans (C/C++) - Manuel Build/Clean/Run Command Setting Not Good Option for C/C++ Development (5)



## Netbeans (C/C++) - Manuel Build/Clean/Run Command Setting Not Good Option for C/C++ Development (6)



## Netbeans (C/C++) - Manuel Build/Clean/Run Command Setting Not Good Option for C/C++ Development (7)



## Netbeans (C/C++) - Manuel Build/Clean/Run Command Setting Not Good Option for C/C++ Development (8)

The image shows a screenshot of the NetBeans IDE's 'Project Properties' dialog for a C/C++ project. The dialog is titled 'Project Properties' and has a 'Project' tab selected. The 'Build' section is visible, showing four input fields for 'Configuration Name', 'Build', 'Clean', and 'Run'. The 'Configuration Name' field is currently empty. The 'Build', 'Clean', and 'Run' fields are also empty. At the bottom of the dialog, there are four buttons: '< Back', 'Next >', 'Finish', and 'Cancel'. The 'Finish' button is highlighted with a blue border.

Project Properties

Project

Configuration Name:

Build:

Clean:

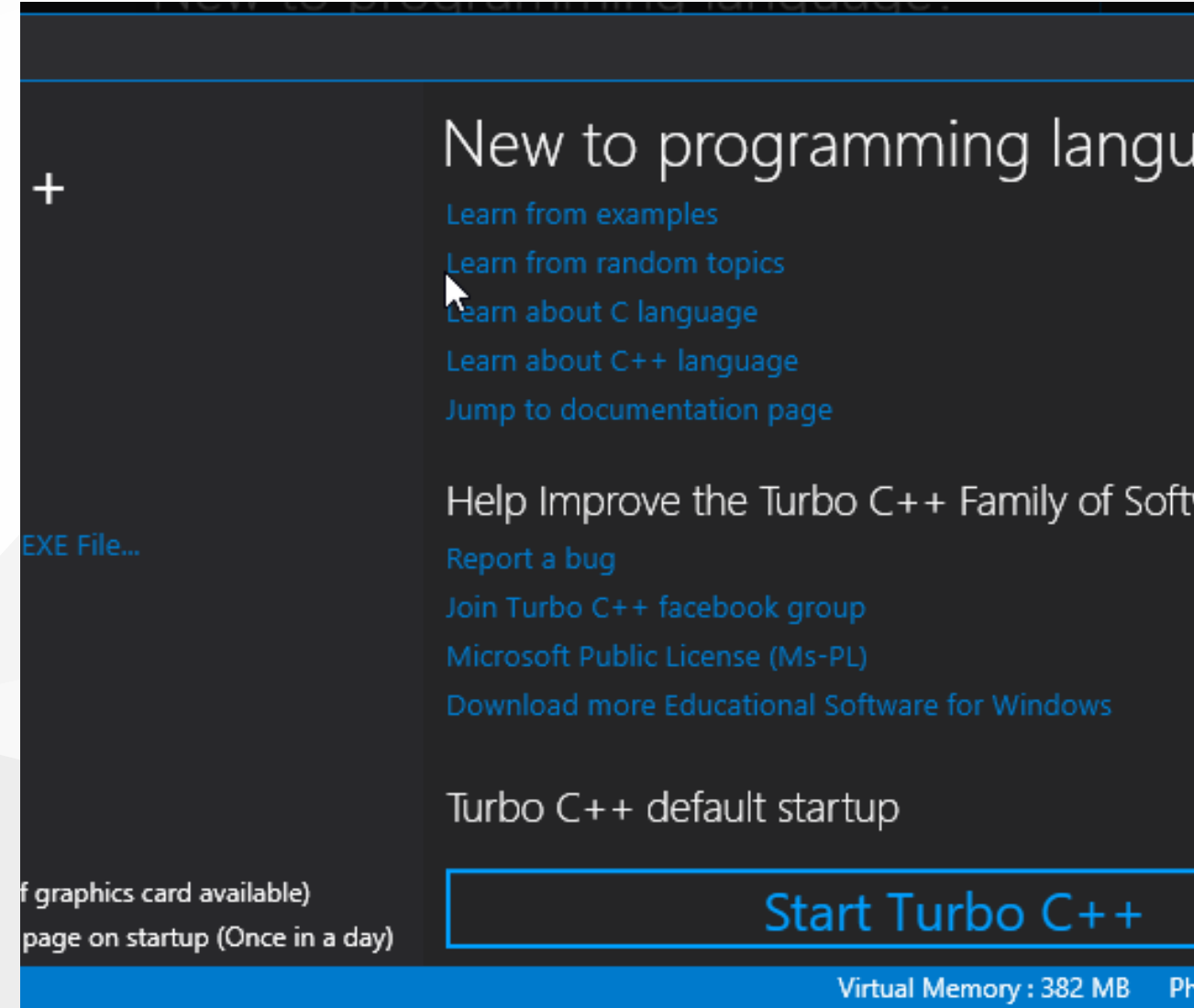
Run:

< Back Next > Finish Cancel

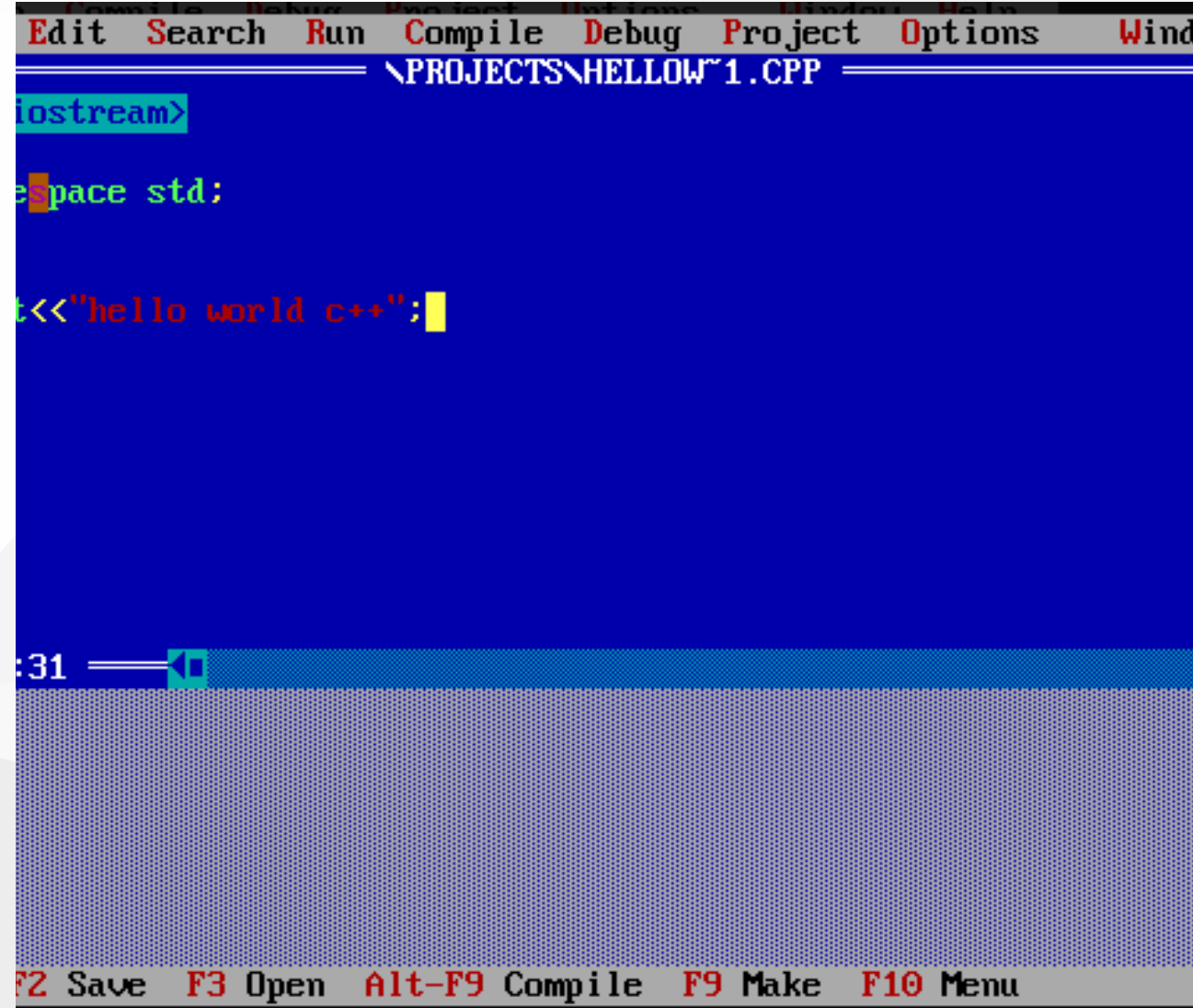
## Turbo C/C++ (1)

Download [Turbo.C.3.2.zip](#)

- Download Turbo C++ for Windows 7, 8, 8.1, 10 and Windows 11 (32-64 bit) with full/window screen mode and many more extra features
- Turbo C++ Shortcuts - C Programming Language Tutorials



## Turbo C/C++ (2)



The screenshot shows the Turbo C++ IDE interface. The menu bar at the top includes 'Edit', 'Search', 'Run', 'Compile', 'Debug', 'Project', 'Options', and 'Wind'. The title bar indicates the file path is '\PROJECTS\HELLOW~1.CPP'. The main editing area has a blue background and contains the following C++ code:

```
iostream>  
using namespace std;  
int main() {  
    cout << "hello world c++";  
}
```

The cursor is positioned at the end of the last line. The status bar at the bottom shows keyboard shortcuts: 'F2 Save', 'F3 Open', 'Alt-F9 Compile', 'F9 Make', and 'F10 Menu'. The line number ':31' is visible on the left side of the editor.



## Cmake (C++/C) (1)

CMake (<http://www.cmake.org/>) is a program which generates the **Makefiles** used by **Make**.

## Cmake (C++/C) (2)

### Why use CMake ?

- Eases **Make** use
  - but the same way of thinking
  - generate the **Makefile**
- Separate the compilation from the sources
- Multi-platfoms
- Very flexible

## Cmake (C++/C) (3)

- Check if the libraries/programs are available on your system
- File generator (**configure\_file**)
- Calling programs or scripts (**doxygen**)
- One of the new standards

## Cmake (C++/C) (4) (Download and Install)

use the following link for download

[Download | CMake](#)

## Cmake (C++/C) (5) (WSL and Linux Environment)

Hello world with CMake

## Cmake (C++/C) (6) (Windows Environment)

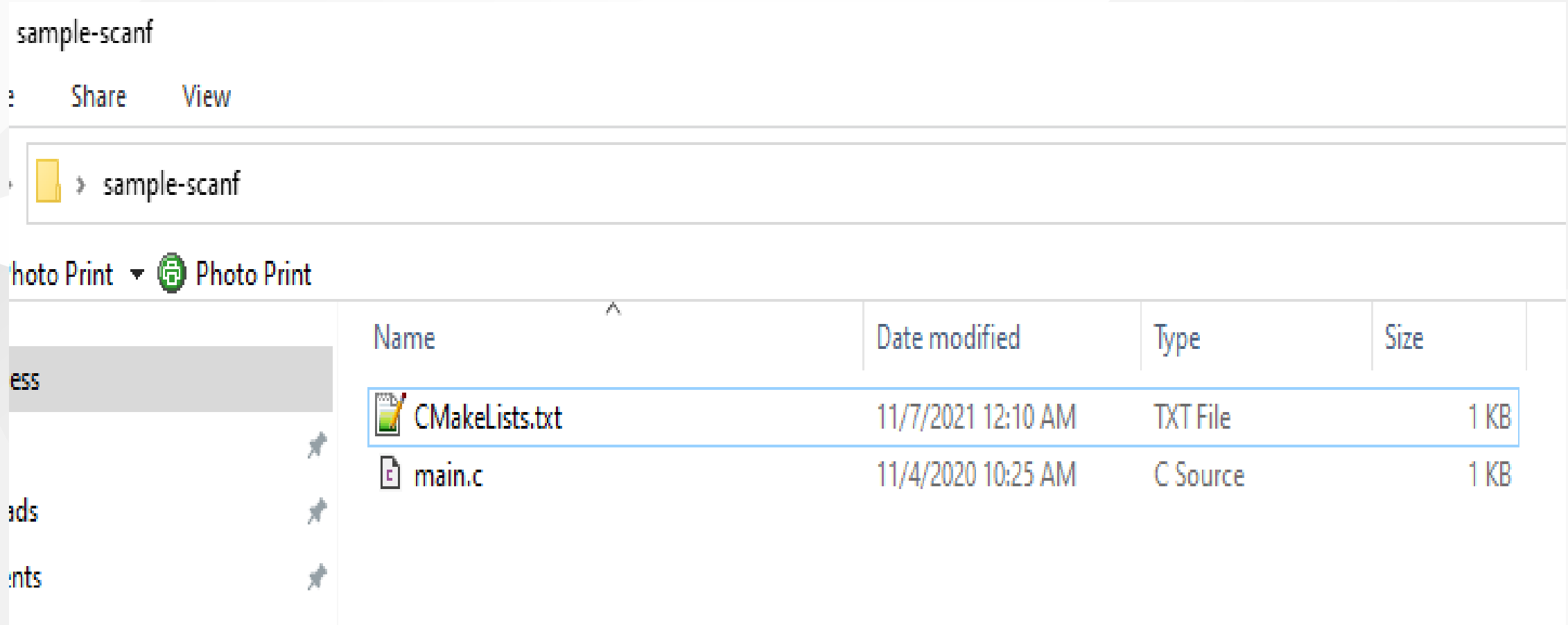
### main.c

```
#include <stdio.h>
int main()
{
    char name[20];
    printf("Enter name: ");
    scanf("%s", name);
    printf("Your name is %s.", name);
    return 0;
}
```

### CMakeLists.txt

```
cmake_minimum_required(VERSION 3.7.2)
project(scanf-sample)
add_executable(scanf-sample main.c)
```

put main.c and CMakeLists.txt file in sample-scanf folder and from command line



run the following cmake command with dot (.) to create solution file for c project

```
C:\Users\ugur.coruh\Desktop\sample-scanf>cmake .
```

## Cmake (C++/C) (8) (Windows Environment)

I have Visual Studio 2022 Community Edition Installed on My Computer, for these reason build tools are selected for visual studio environment and the following outputs are generated

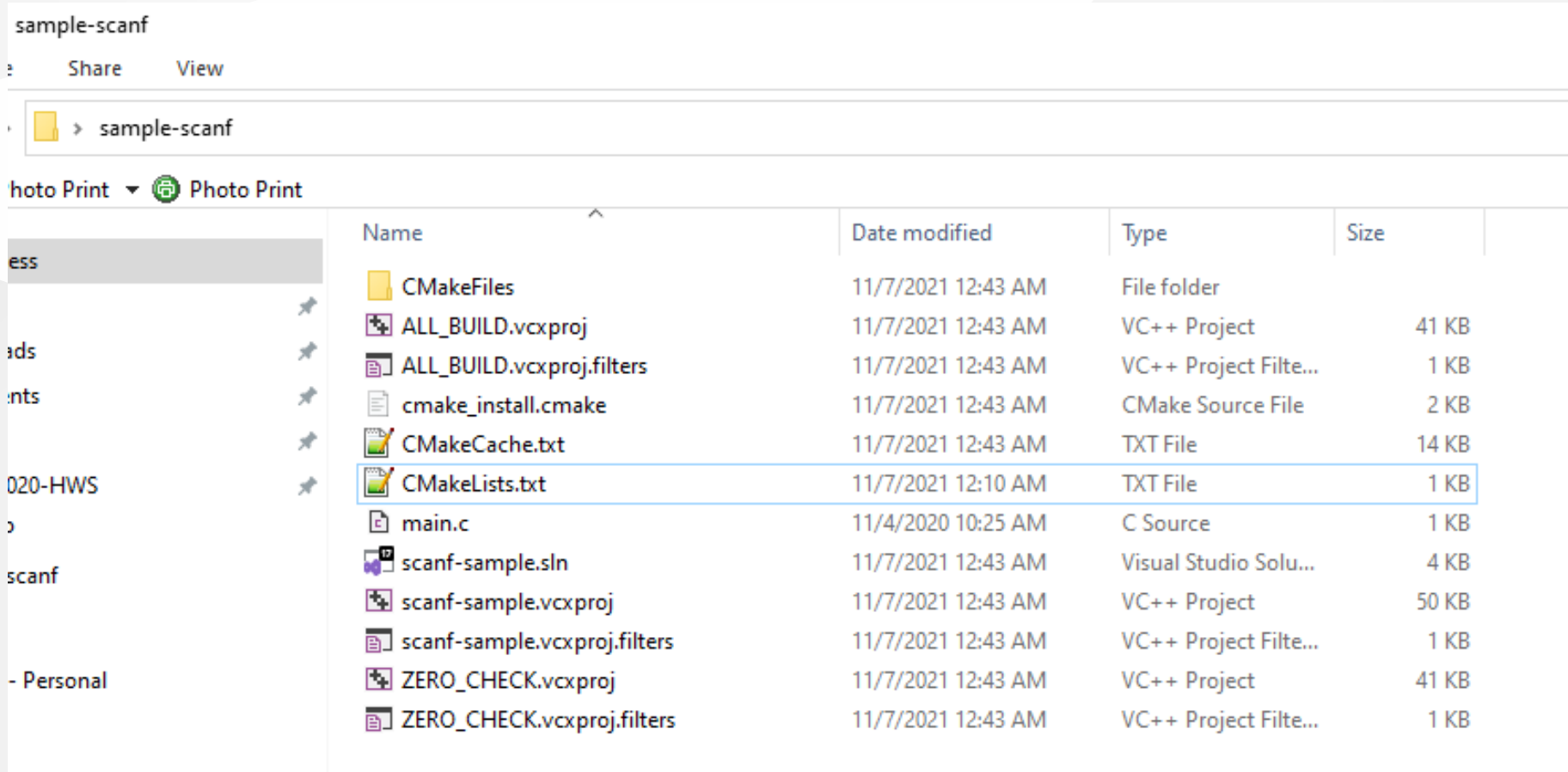
```
C:\Users\ugur.coruh\Desktop\sample-scanf>cmake .
-- Building for: Visual Studio 17 2022
-- Selecting Windows SDK version 10.0.22000.0 to target Windows 10.0.19043.
-- The C compiler identification is MSVC 19.30.30704.0
-- The CXX compiler identification is MSVC 19.30.30704.0
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working C compiler: C:/Program Files/Microsoft Visual Studio/2022/Community/VC/Tools/MSVC/14.30.30704/bin/Hostx64/x64/cl.exe - skipped
-- Detecting C compile features
-- Detecting C compile features - done
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Check for working CXX compiler: C:/Program Files/Microsoft Visual Studio/2022/Community/VC/Tools/MSVC/14.30.30704/bin/Hostx64/x64/cl.exe - skipped
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Configuring done
-- Generating done
-- Build files have been written to: C:/Users/ugur.coruh/Desktop/sample-scanf

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



## Cmake (C++/C) (9) (Windows Environment)

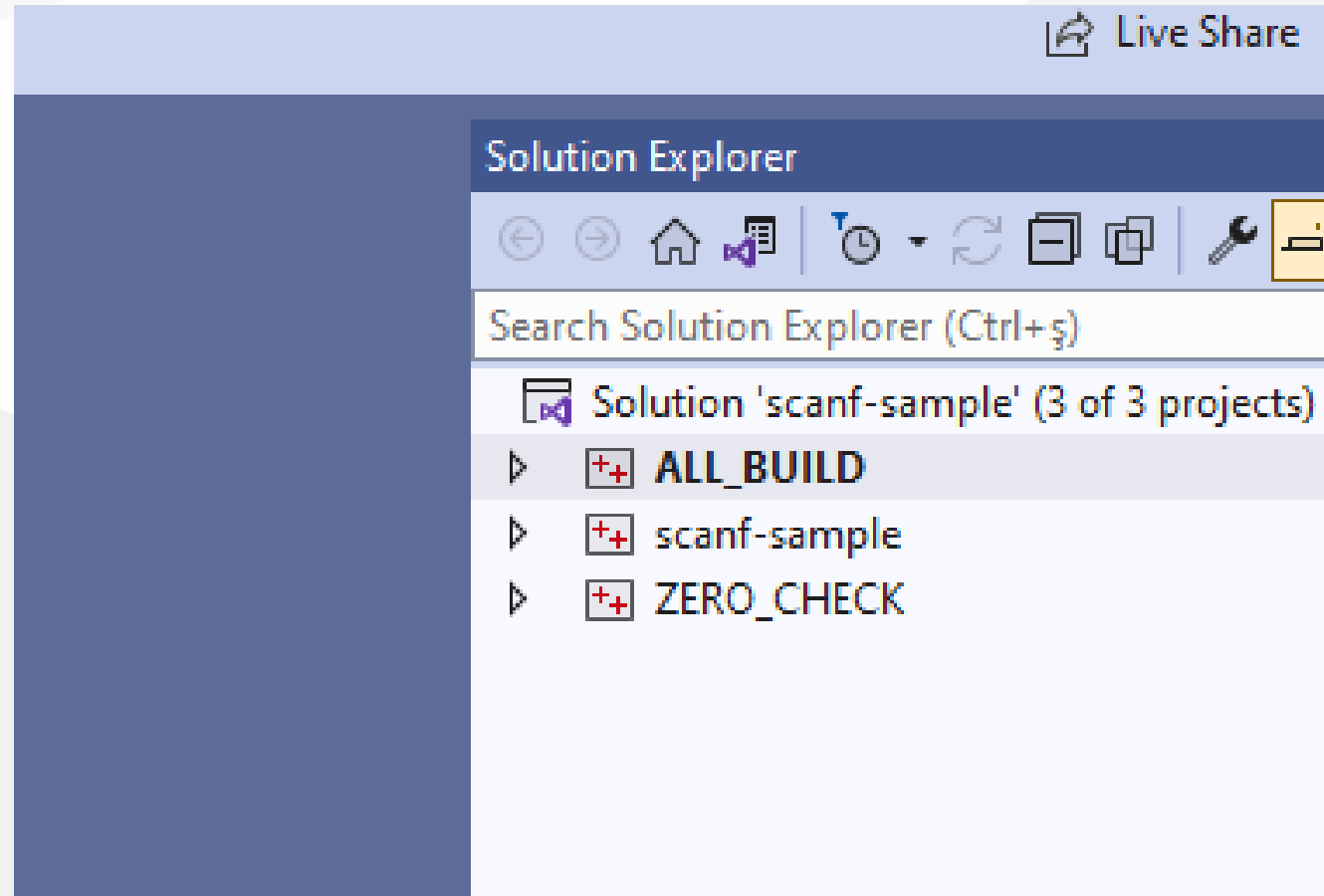
also following files are generated



Name	Date modified	Type	Size
CMakeFiles	11/7/2021 12:43 AM	File folder	
ALL_BUILD.vcxproj	11/7/2021 12:43 AM	VC++ Project	41 KB
ALL_BUILD.vcxproj.filters	11/7/2021 12:43 AM	VC++ Project Filte...	1 KB
cmake_install.cmake	11/7/2021 12:43 AM	CMake Source File	2 KB
CMakeCache.txt	11/7/2021 12:43 AM	TXT File	14 KB
CMakeLists.txt	11/7/2021 12:10 AM	TXT File	1 KB
main.c	11/4/2020 10:25 AM	C Source	1 KB
scanf-sample.sln	11/7/2021 12:43 AM	Visual Studio Solu...	4 KB
scanf-sample.vcxproj	11/7/2021 12:43 AM	VC++ Project	50 KB
scanf-sample.vcxproj.filters	11/7/2021 12:43 AM	VC++ Project Filte...	1 KB
ZERO_CHECK.vcxproj	11/7/2021 12:43 AM	VC++ Project	41 KB
ZERO_CHECK.vcxproj.filters	11/7/2021 12:43 AM	VC++ Project Filte...	1 KB

## Cmake (C++/C) (10) (Windows Environment)

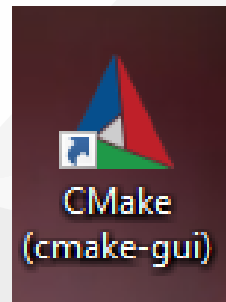
if we open scanf-sample.sln file we will have automated generated project files



## Cmake (C++/C) (11) (Windows Environment)

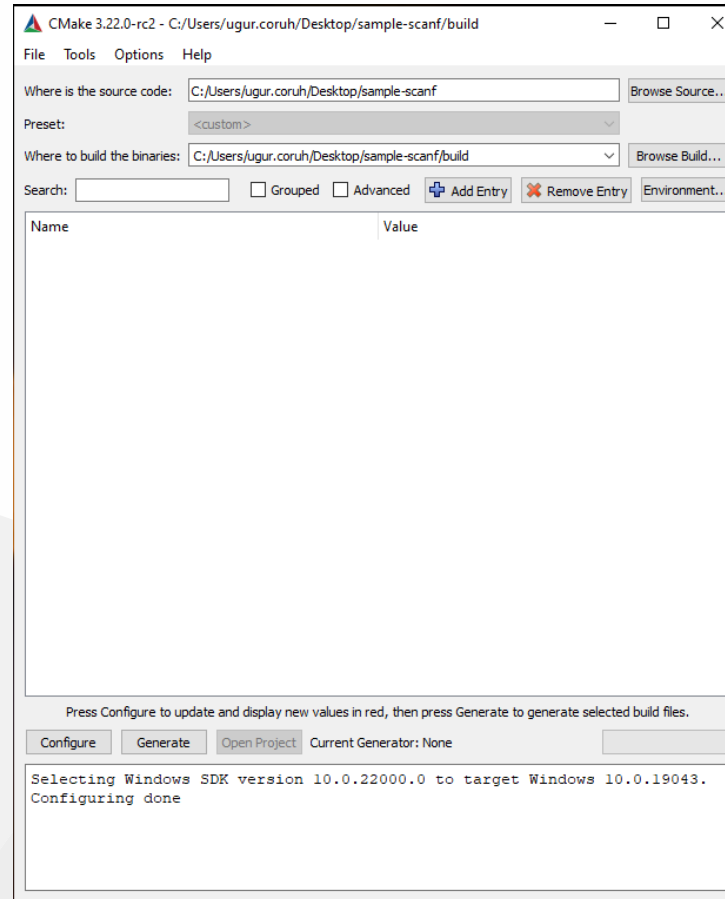
you can make scanf-sample with startup project with right click and then run on visual studio.

if you want to configure for another build tool you can use Cmake-GUI installed with setup on your computer



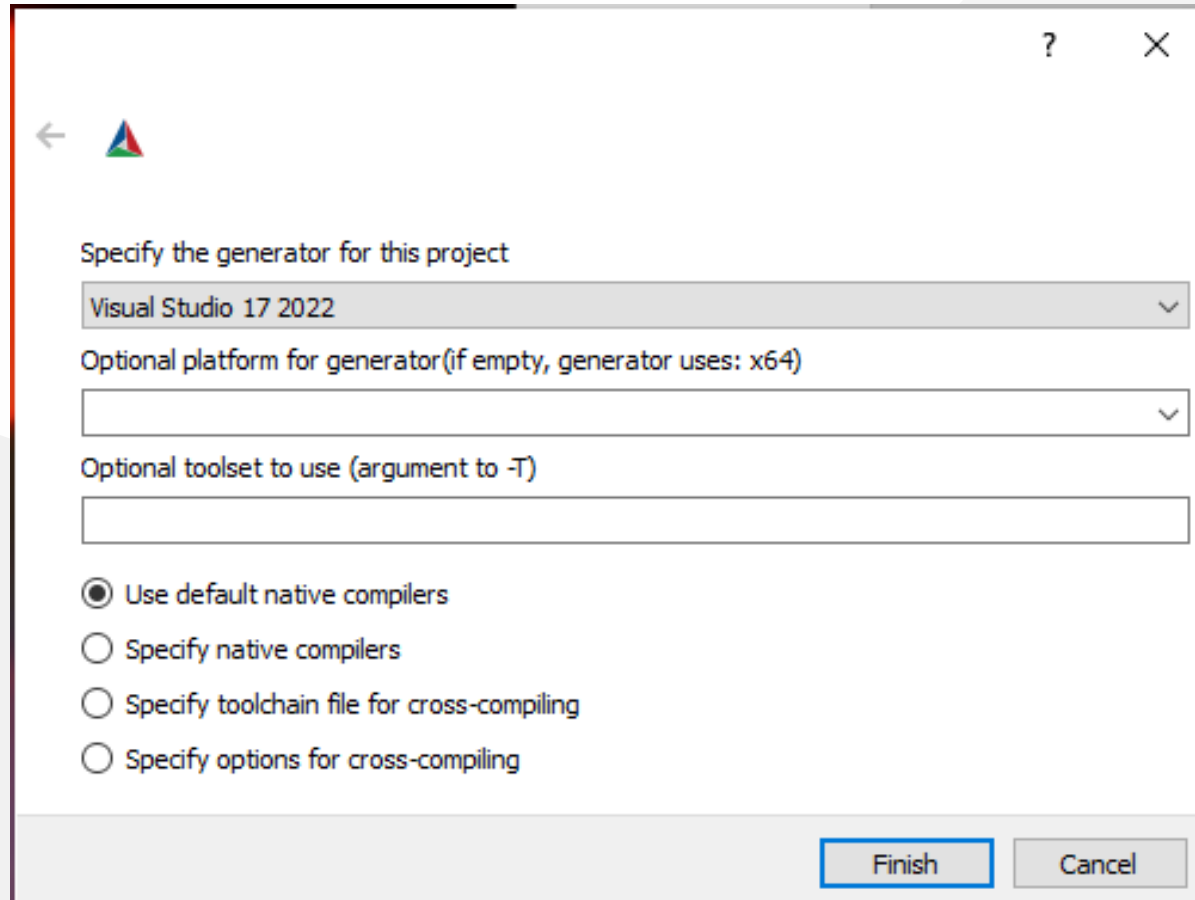
## Cmake (C++/C) (12) (Windows Environment)

Open GUI and Select *File-> Delete Cache*

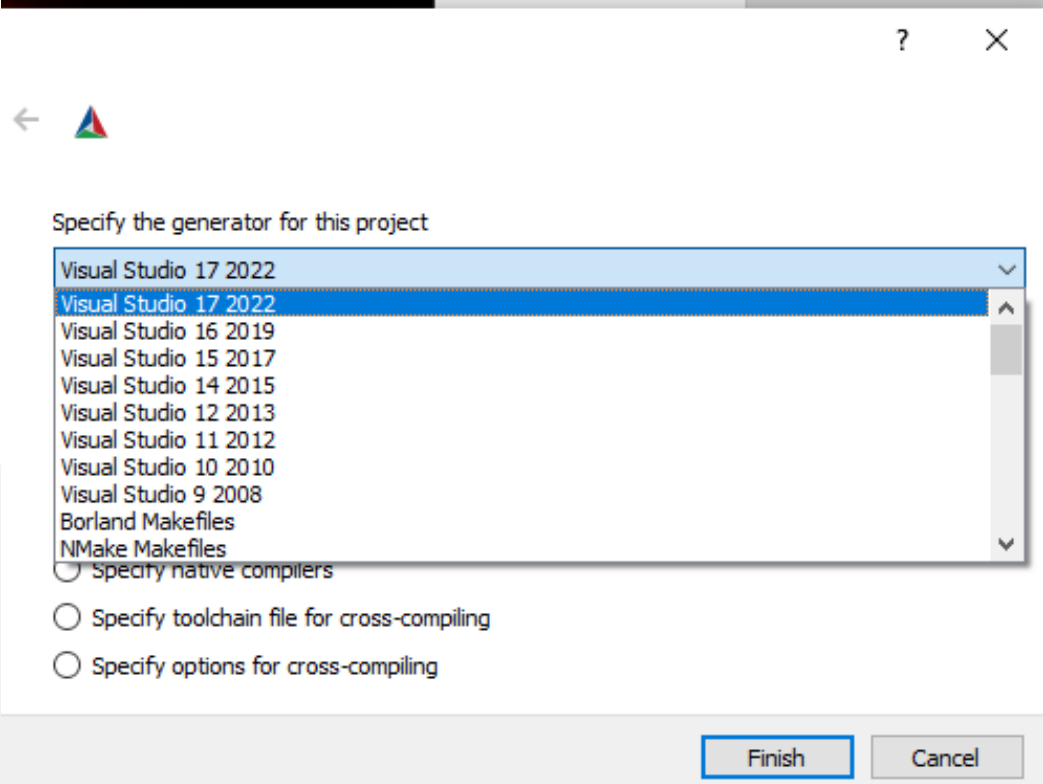


## Cmake (C++/C) (13) (Windows Environment)

then you can click "Configure" to select build tool



# Cmake (C++/C) (14) (Windows Environment)



## Cmake (C++/C) (15) (Windows Environment)

if you click "Configure" twice it will generate the visual studio solution in build folder

for more detailed examples that include also docker and travis-ci sample you can check the following repo

[GitHub - ttroy50/cmake-examples: Useful CMake Examples](#)

## Make (1)

Sample

hello.c

```
#include <stdio.h>

int main(void)
{
    printf("hello, world\n");
}
```



## Make (2)

### Makefile

```
# This is the default target, which will be built when
# you invoke make
.PHONY: all
all: hello

# This rule tells make how to build hello from hello.cpp
hello: hello.c
    g++ -o hello hello.c

# This rule tells make to copy hello to the binaries subdirectory,
# creating it if necessary
.PHONY: install
install:
    mkdir -p binaries
    cp -p hello binaries

# This rule tells make to delete hello and hello.o
.PHONY: clean
clean:
    rm -f hello
```

## Make (3)

### compile.bat

```
make all .
```

will create hello.exe

check hello-make sample

# Make (4)

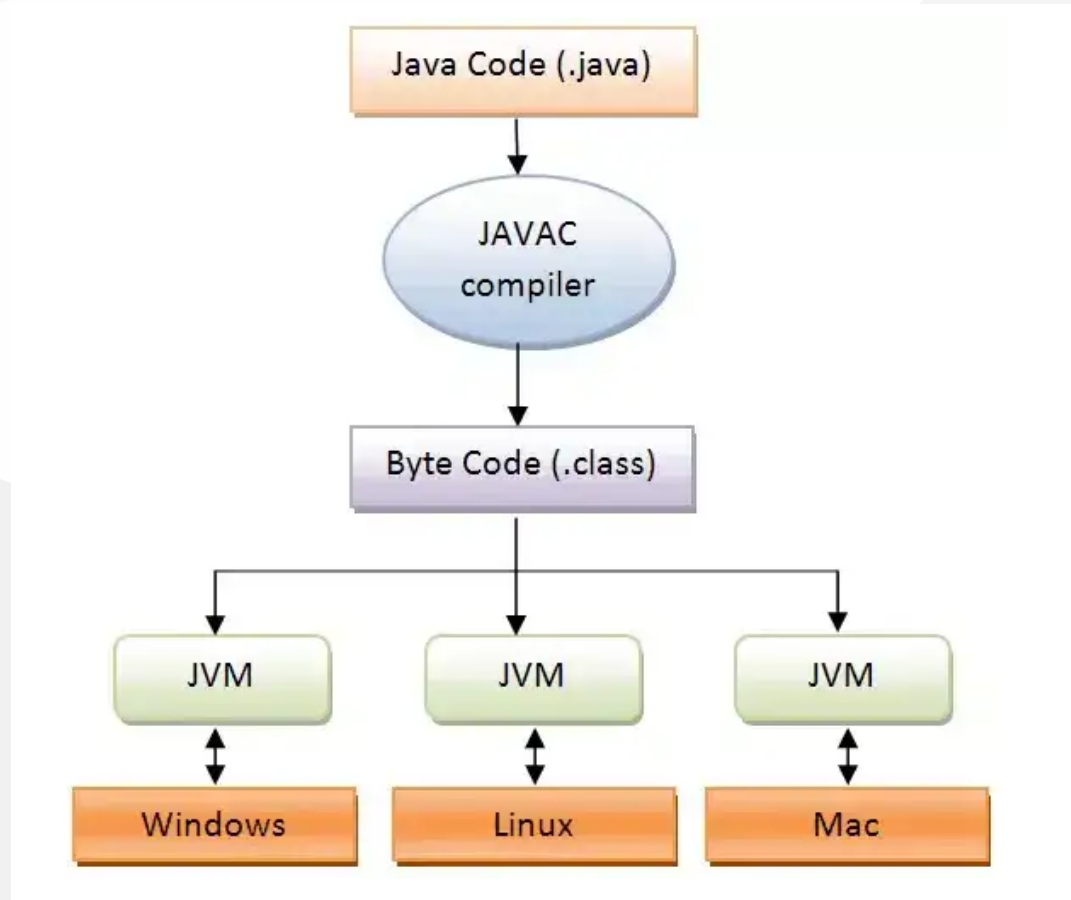
s-and-programming-I > Week-2 > hello-make

Print

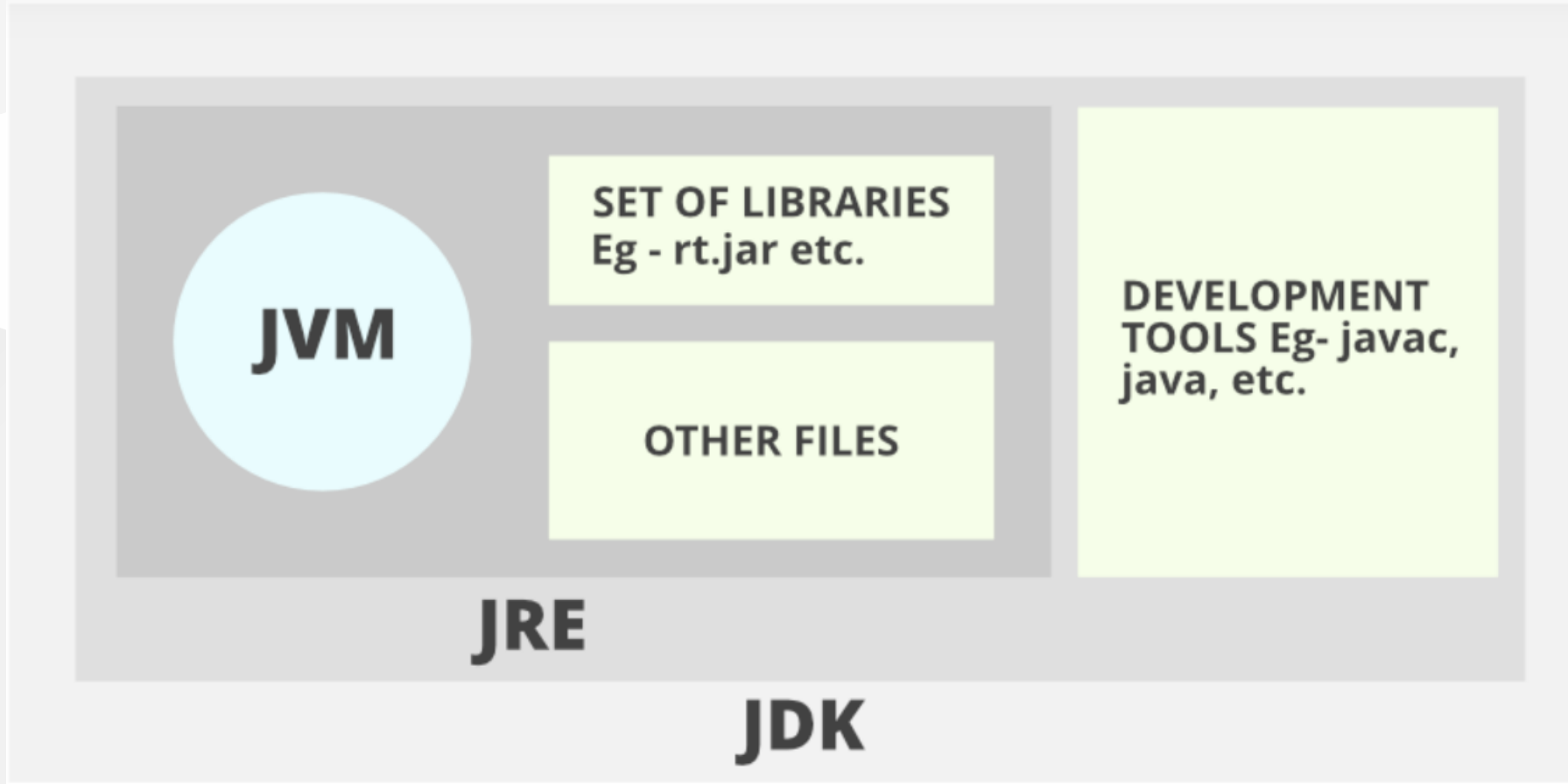
	Name	Date modified
	compile.bat	11/7/2021
	hello.c	11/2/2021
	Makefile	11/2/2021

ing-I

# JAVA Environment and Development



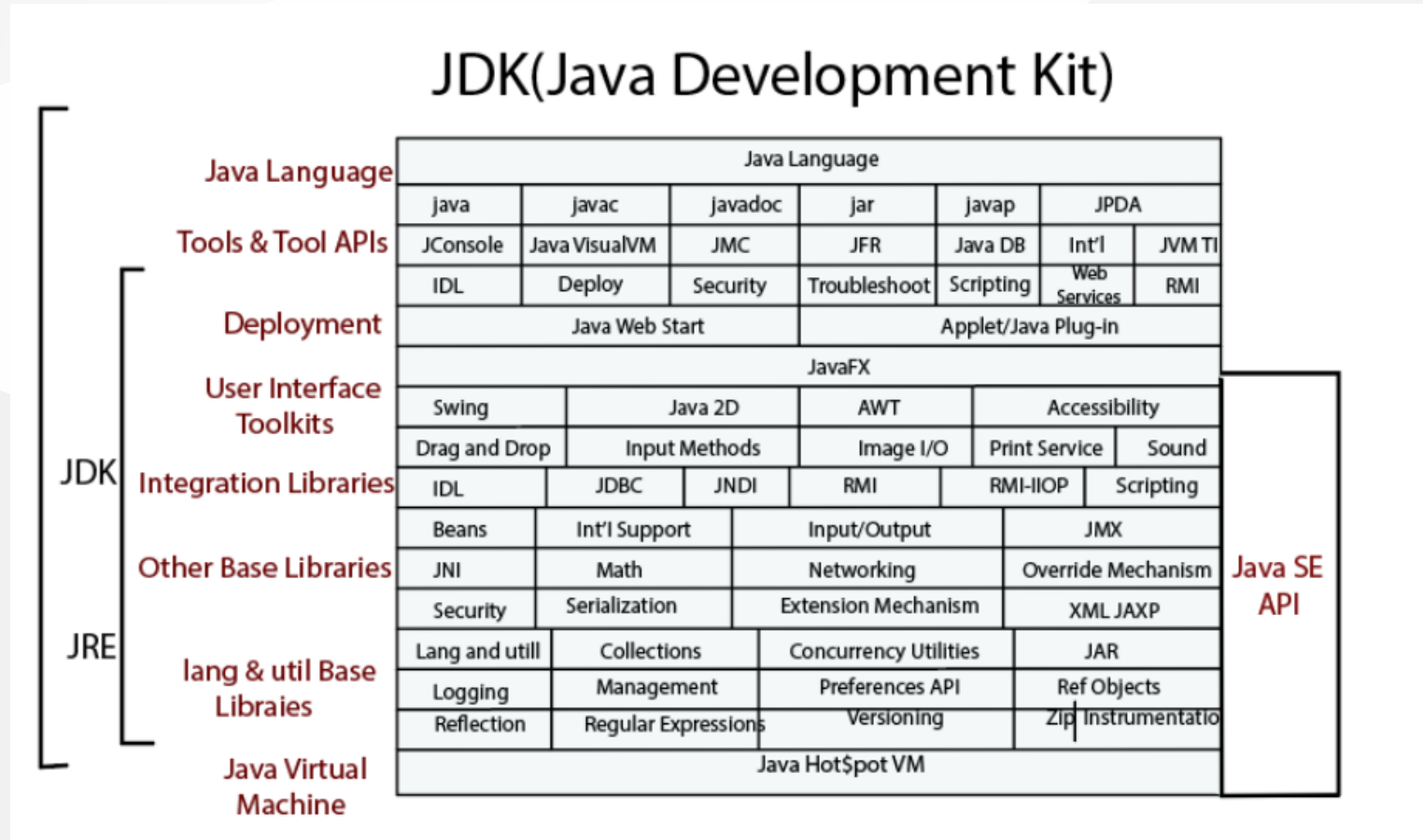
## JDK and JRE Setup (1)



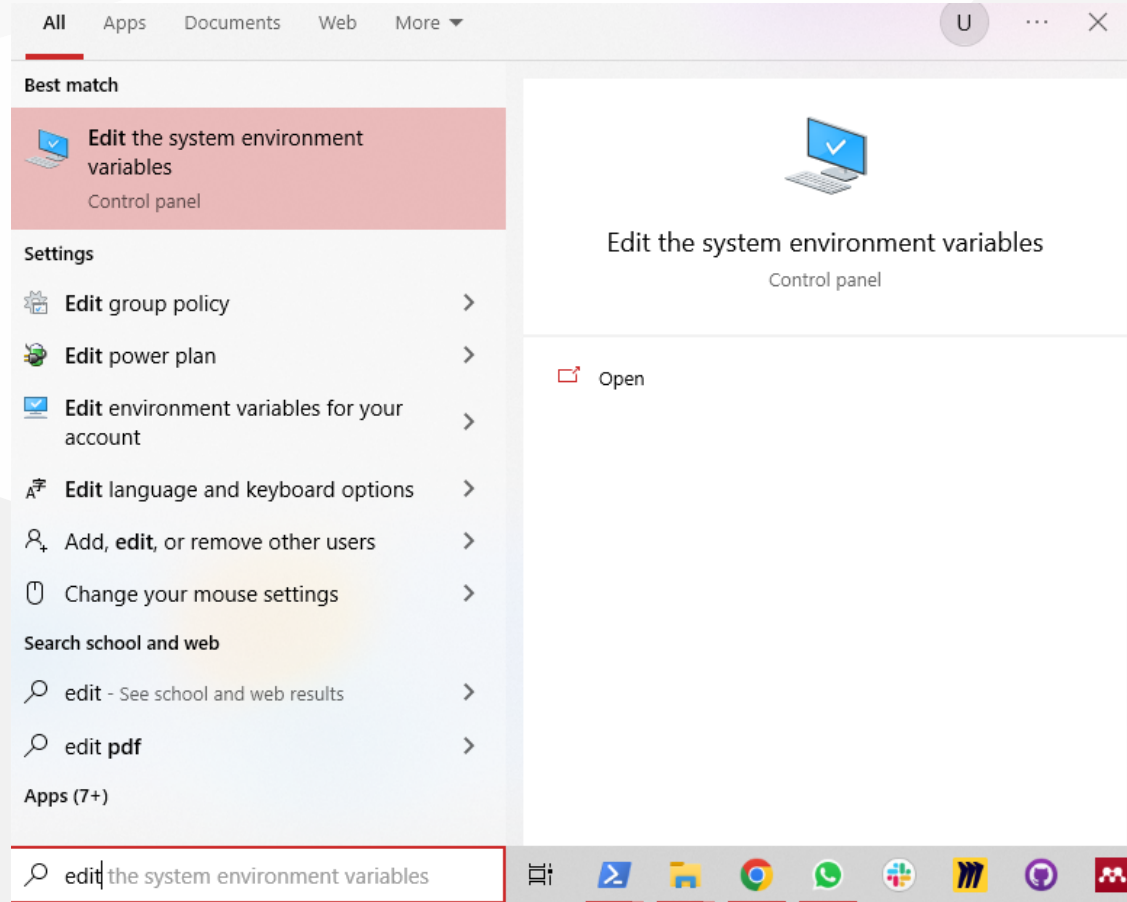
## JDK and JRE Setup (2)

- **JDK** (Java Development Kit) is a Kit that provides the environment to **develop and execute(run)** the Java program. JDK is a kit(or package) that includes two things
  - Development Tools(to provide an environment to develop your java programs)
  - JRE (to execute your java program).
- **JRE** (Java Runtime Environment) is an installation package that provides an environment to **only run(not develop)** the java program(or application)onto your machine. JRE is only used by those who only want to run Java programs that are end-users of your system.
- **JVM (Java Virtual Machine)** is a very important part of both JDK and JRE because it is contained or inbuilt in both. Whatever Java program you run using JRE or JDK goes into JVM and JVM is responsible for executing the java program line by line, hence it is also known as an **\*\*j\*\*\*\*nterpreter\*\***.

- Difference between JDK, JRE, JVM - TutorialAndExample



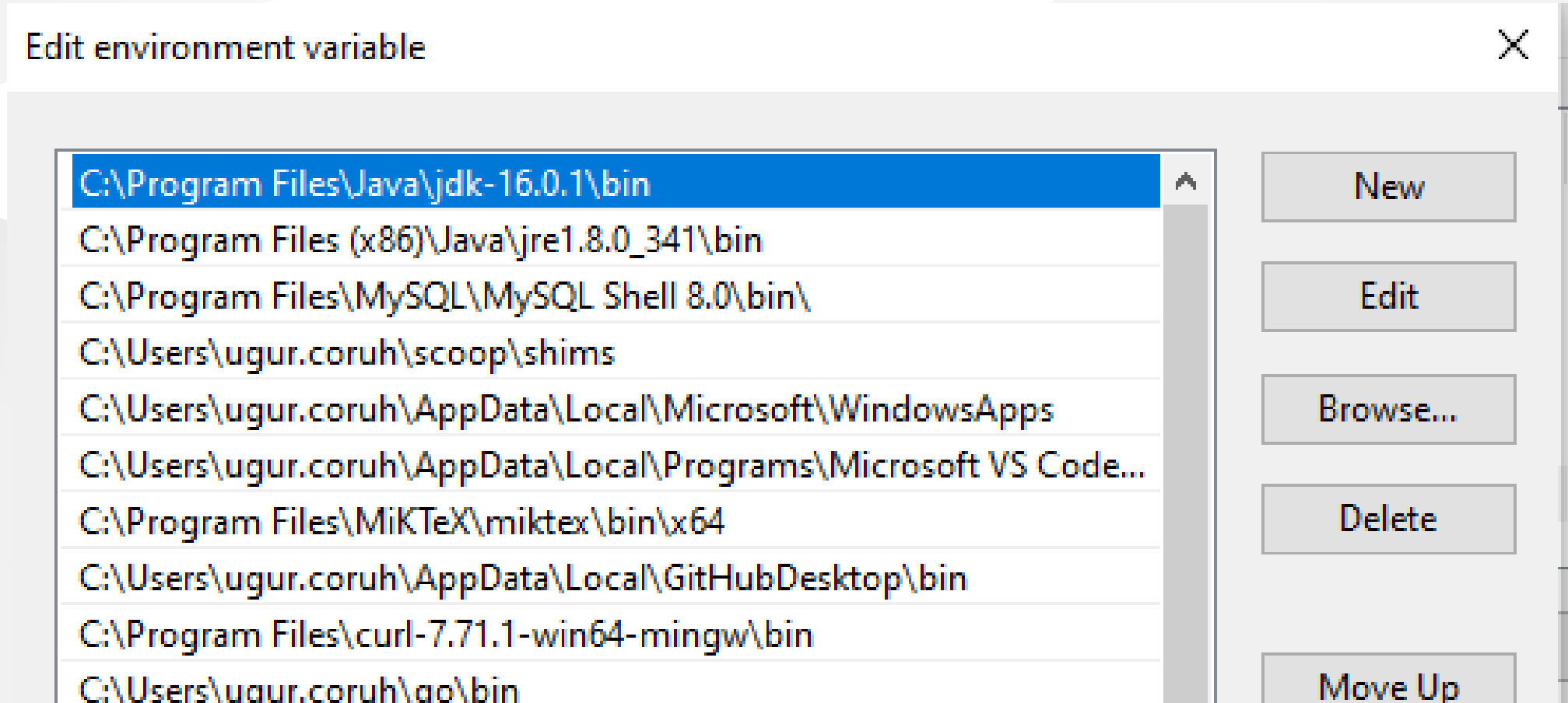
# System Environments and Paths for Java (1)





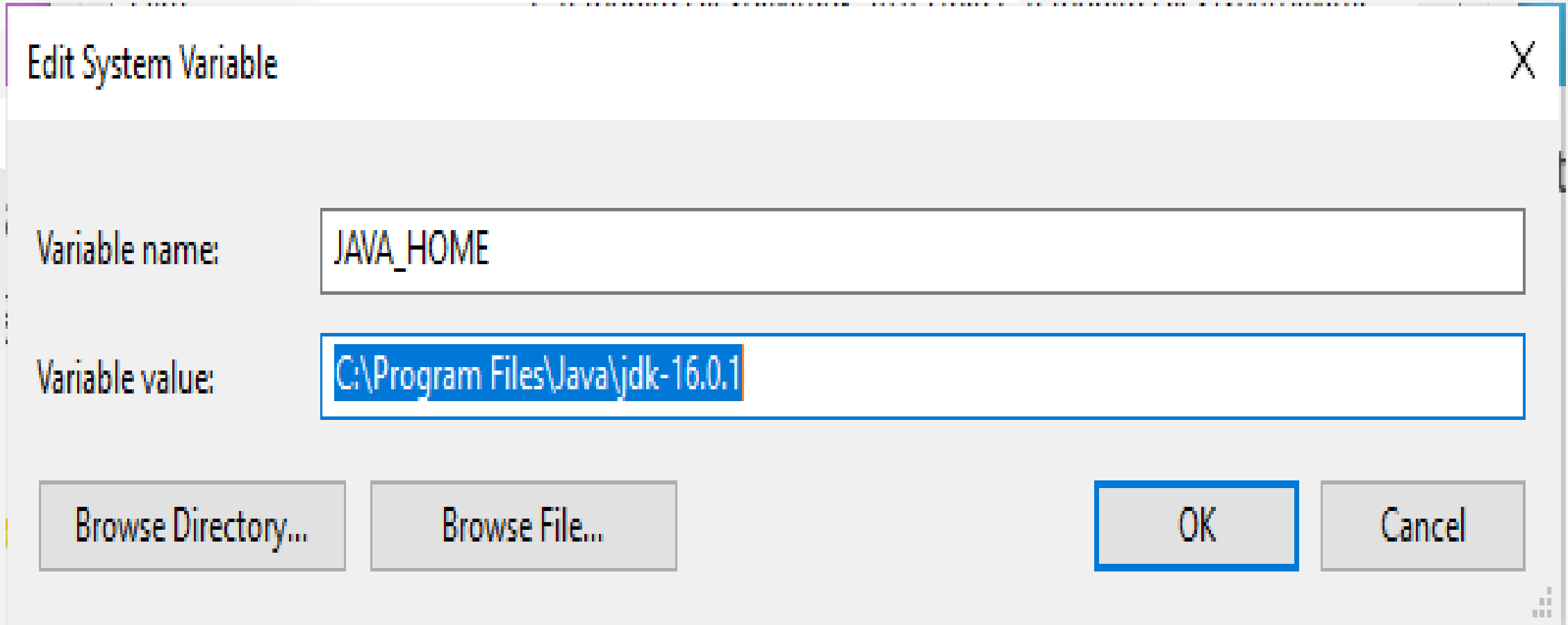
## System Environments and Paths for Java (2)

- Select path variable (JDK should be set there)



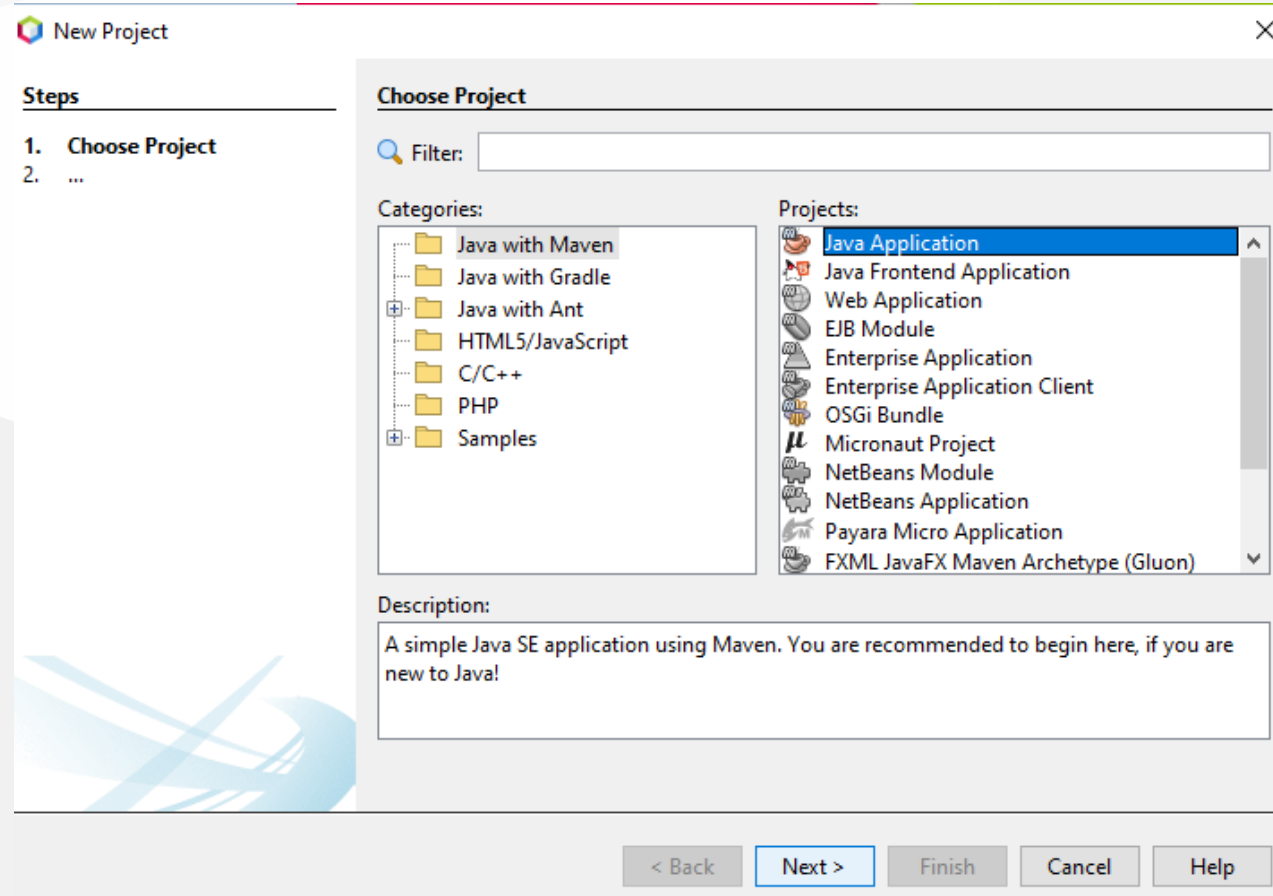
## System Environments and Paths for Java (3)

- JAVA\_HOME also should be set

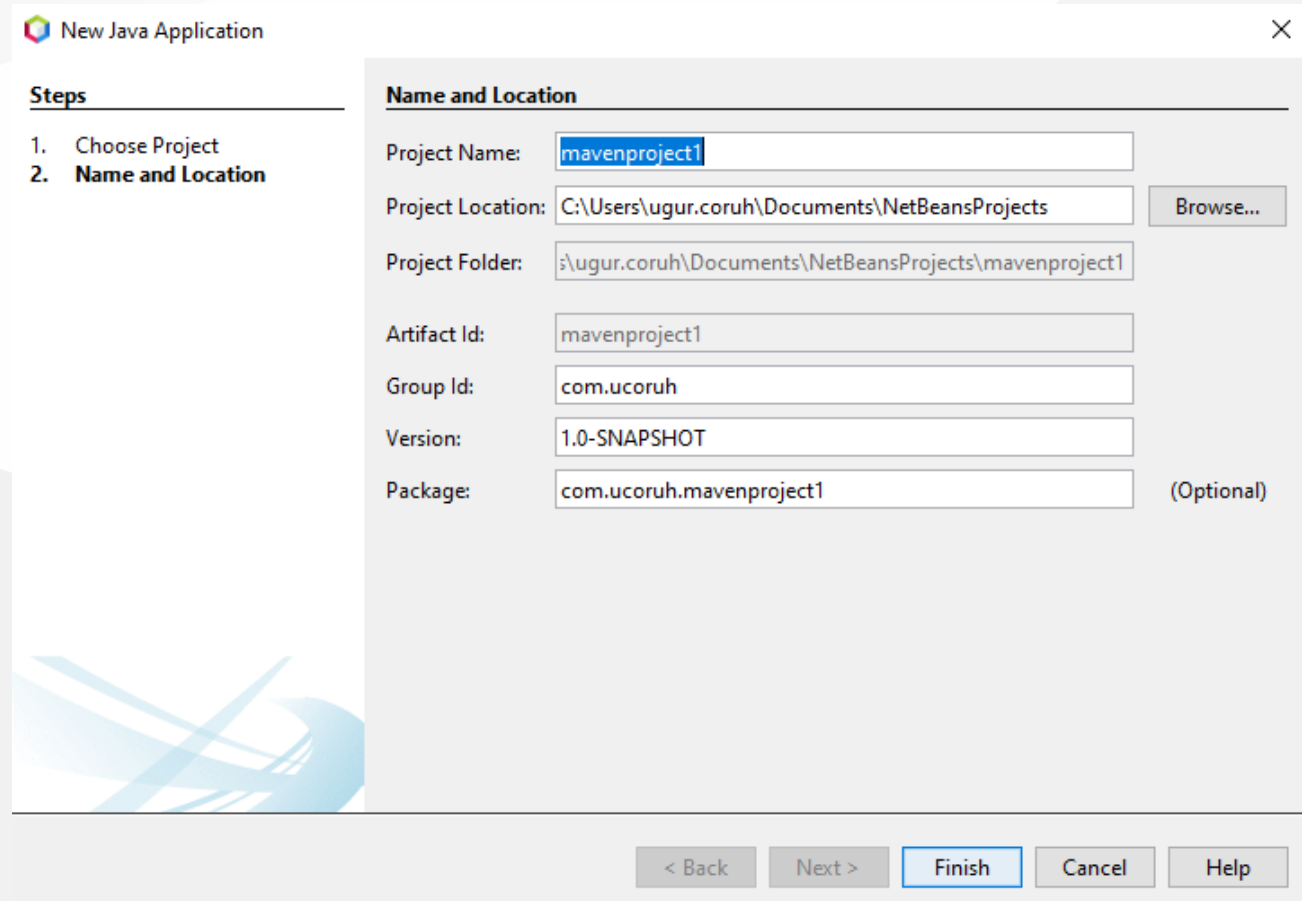


## Netbeans (Java) (1)

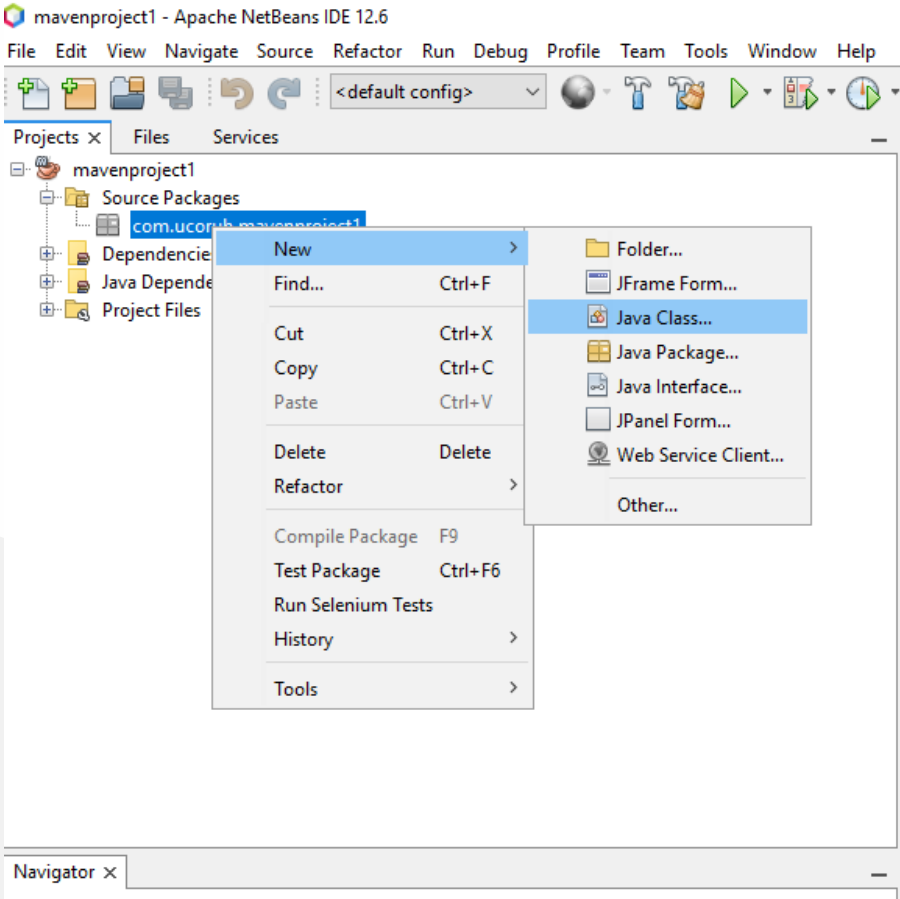
- Open New Project -> Java Project



## Netbeans (Java) (2)



# Netbeans (Java) (3)



## Netbeans (Java) (4)

**New Java Class**

**Steps**

1. Choose File Type
2. **Name and Location**

**Name and Location**

Class Name:

Project:

Location:

Package:

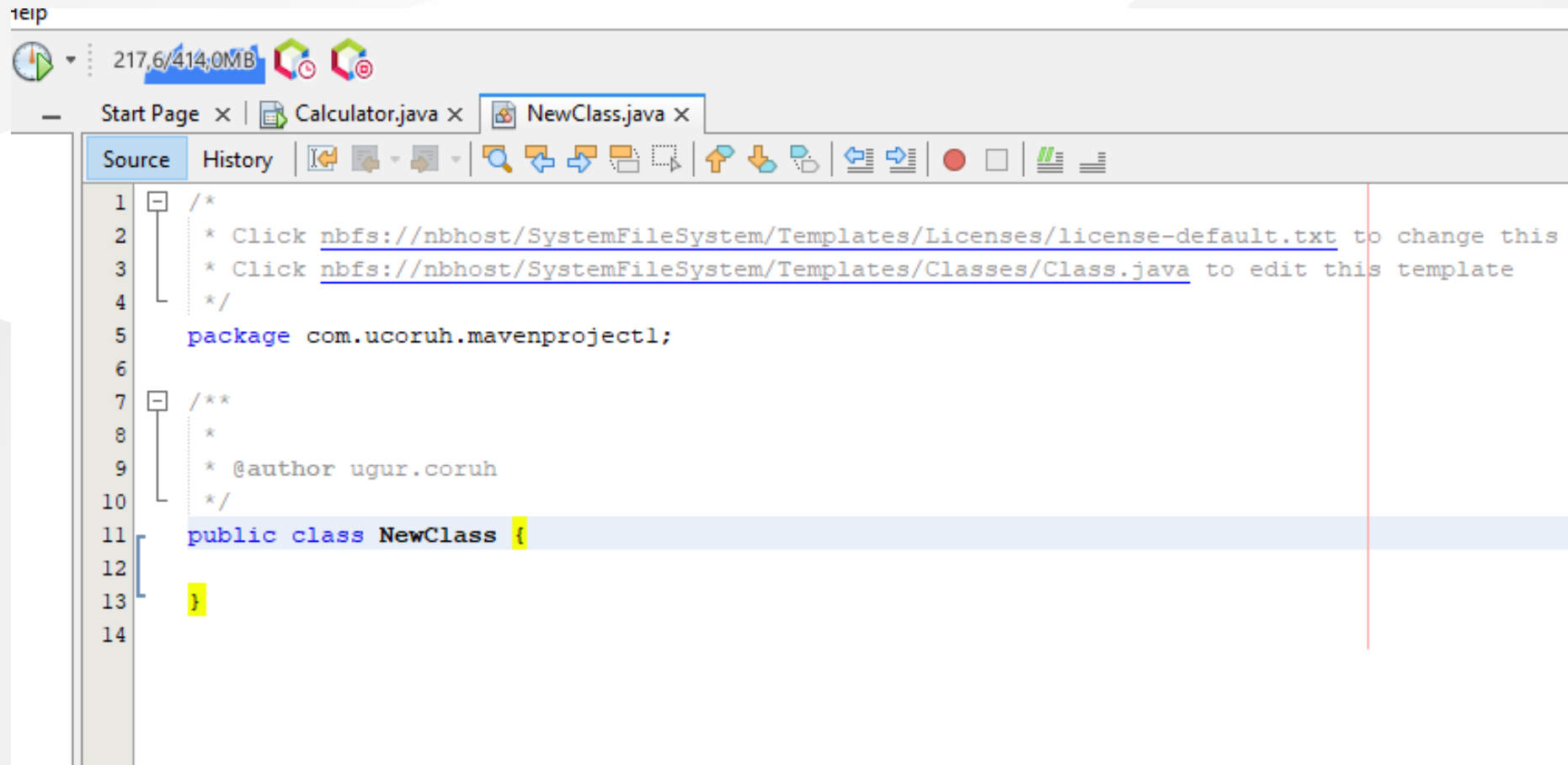
Created File:

Superclass:

Interfaces:

< Back   Next >   **Finish**   Cancel   Help

## Netbeans (Java) (5)



The screenshot shows the NetBeans IDE interface. The top bar displays system information: '1eip', a clock, and memory usage '217,6/414,0MB'. The window title bar shows 'Start Page x | Calculator.java x | NewClass.java x'. The 'Source' tab is active, showing the following code:

```
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4   */
5  package com.ucoruh.mavenproject1;
6
7  /**
8   *
9   * @author ugur.coruh
10 */
11 public class NewClass {
12
13 }
14
```

## Update code and run



```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package com.ucoruh.mavenproject1;

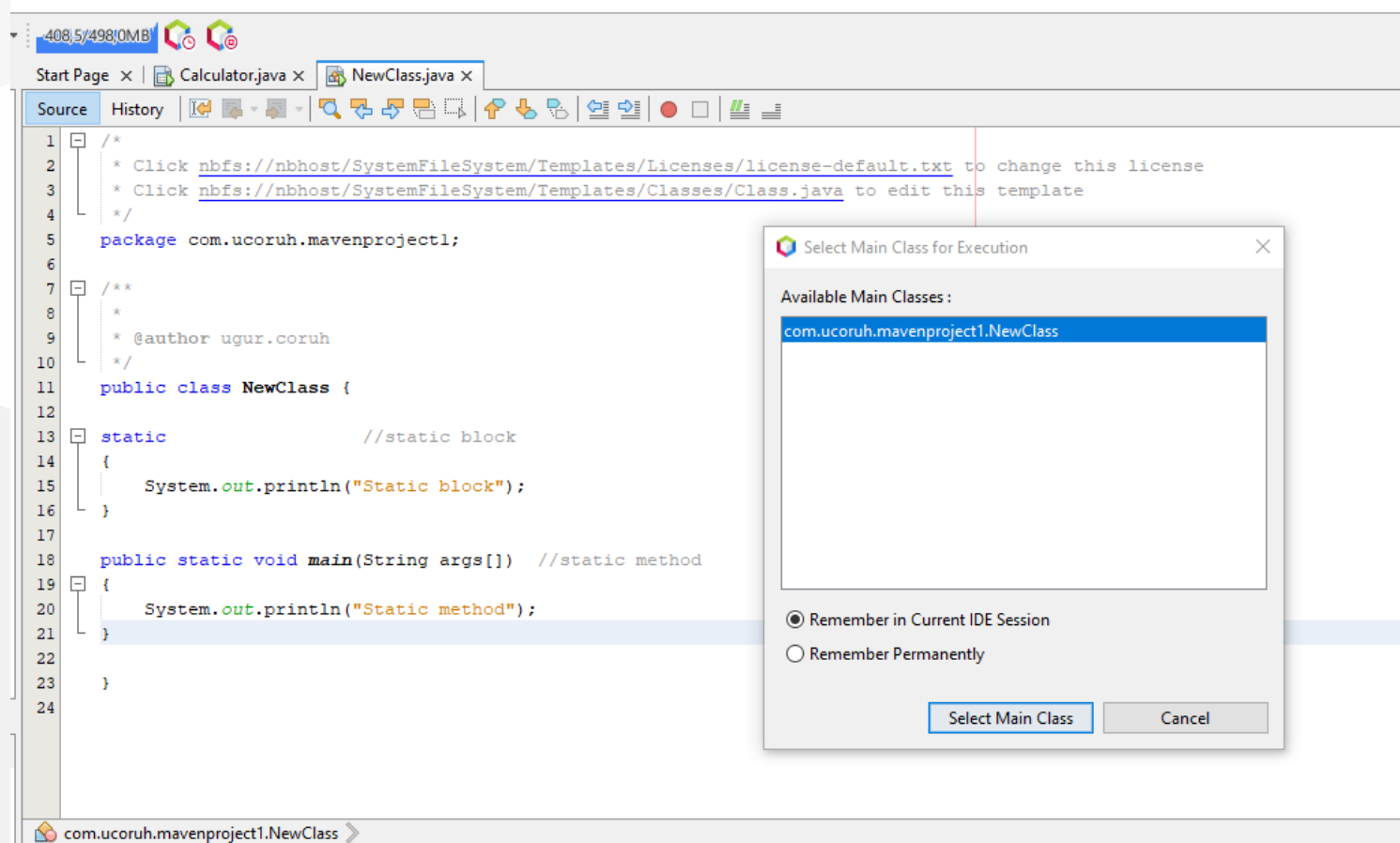
/**
 *
 * @author ugur.coruh
 */
public class NewClass {

    static //static block
    {
        System.out.println("Static block");
    }

    public static void main(String args[]) //static method
    {
        System.out.println("Static method");
    }
}
```



## Netbeans (Java) (7)



The screenshot displays the NetBeans IDE interface. The main editor window shows the source code of a Java class named `NewClass` in the package `com.ucoruh.mavenproject1`. The code includes a static block and a `main` method. A dialog box titled "Select Main Class for Execution" is open, showing the available main classes. The class `com.ucoruh.mavenproject1.NewClass` is selected. The dialog also has radio buttons for "Remember in Current IDE Session" (selected) and "Remember Permanently".

```
1  /*
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3  * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4  */
5  package com.ucoruh.mavenproject1;
6
7  /**
8   *
9   * @author ugor.coruh
10  */
11  public class NewClass {
12
13  static          //static block
14  {
15      System.out.println("Static block");
16  }
17
18  public static void main(String args[]) //static method
19  {
20      System.out.println("Static method");
21  }
22
23  }
24
```

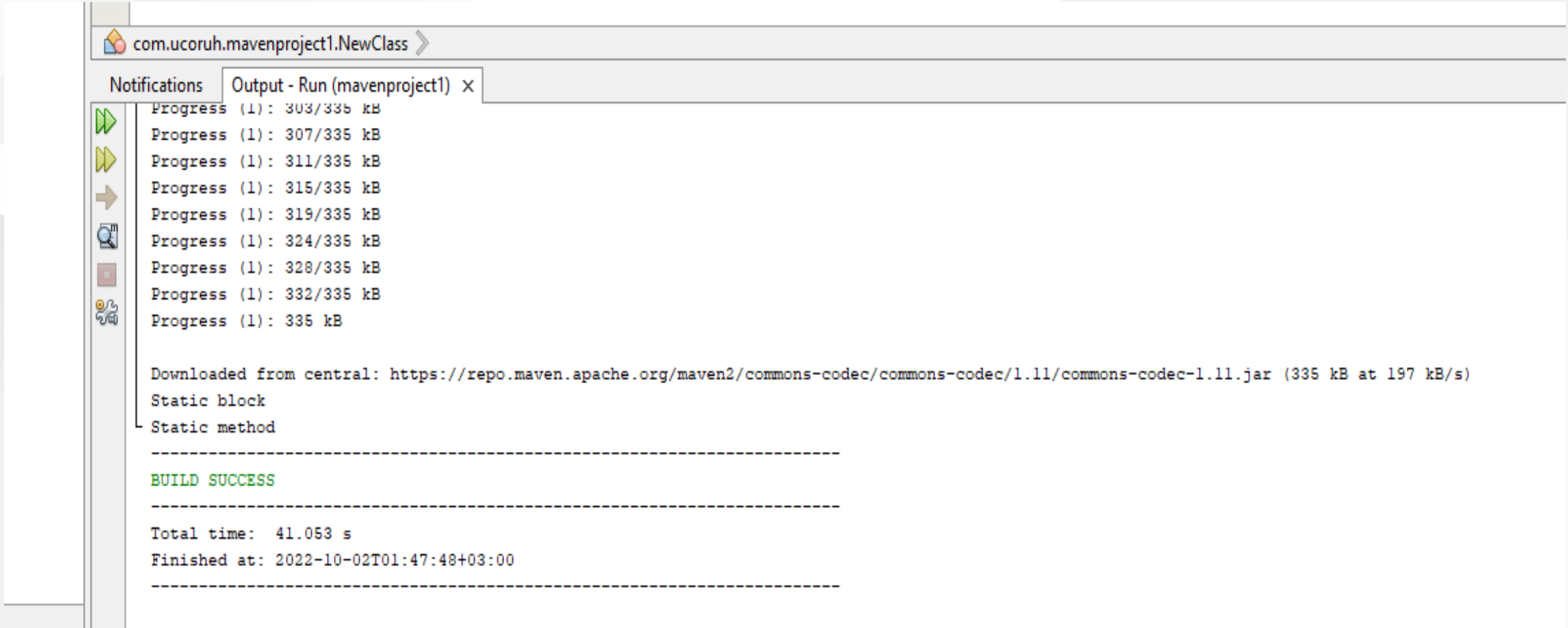
Available Main Classes :

- `com.ucoruh.mavenproject1.NewClass`

Remember in Current IDE Session  
 Remember Permanently

Select Main Class Cancel

## Netbeans (Java) (8)

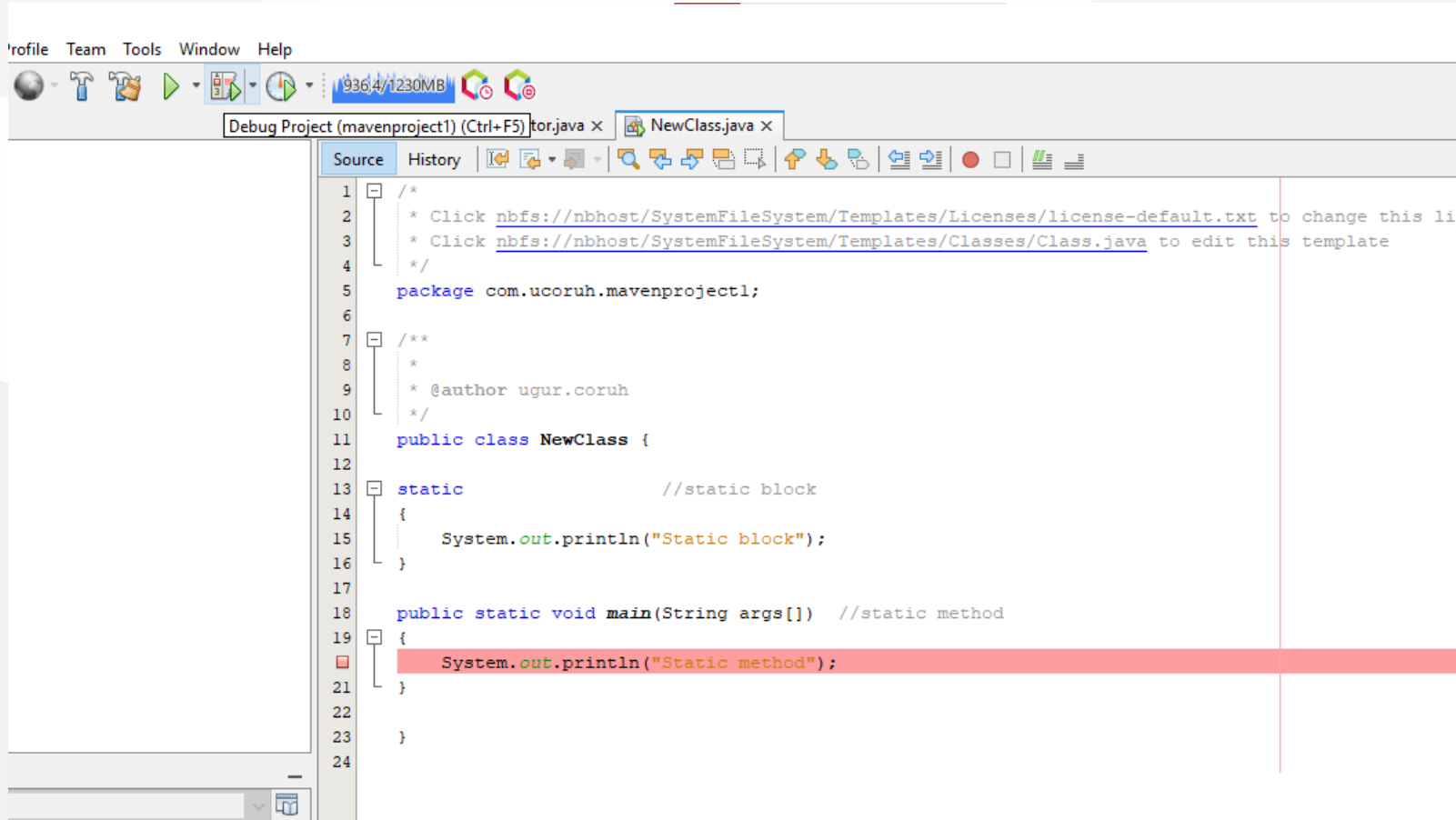


```
com.ucoruh.mavenproject1.NewClass >
Notifications Output - Run (mavenproject1) x
Progress (1): 303/335 kB
Progress (1): 307/335 kB
Progress (1): 311/335 kB
Progress (1): 315/335 kB
Progress (1): 319/335 kB
Progress (1): 324/335 kB
Progress (1): 328/335 kB
Progress (1): 332/335 kB
Progress (1): 335 kB

Downloaded from central: https://repo.maven.apache.org/maven2/commons-codec/commons-codec/1.11/commons-codec-1.11.jar (335 kB at 197 kB/s)
Static block
Static method
-----
BUILD SUCCESS
-----

Total time: 41.053 s
Finished at: 2022-10-02T01:47:48+03:00
-----
```

## Netbeans (Java) (9)



```
1  /*
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this li
3  * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4  */
5  package com.ucoruh.mavenproject1;
6
7  /**
8   *
9   * @author ugur.coruh
10  */
11  public class NewClass {
12
13  static //static block
14  {
15      System.out.println("Static block");
16  }
17
18  public static void main(String args[]) //static method
19  {
20      System.out.println("Static method");
21  }
22
23  }
24
```

# Netbeans (Java) (10)

The screenshot displays the Apache NetBeans IDE 12.6 interface. The main window shows the source code for 'NewClass.java' with the following content:

```
1  /*  
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license  
3  * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template  
4  */  
5  package com.ucooruh.mavenproject1;  
6  
7  /**  
8  *  
9  * @author ugur.coruh  
10 */  
11 public class NewClass {  
12  
13     //static block  
14     {  
15         System.out.println("Static block");  
16     }  
17  
18     public static void main(String args[]) //static method  
19     {  
20         System.out.println("Static method");  
21     }  
22  
23 }  
24
```

The 'Debugger' window shows the following state:

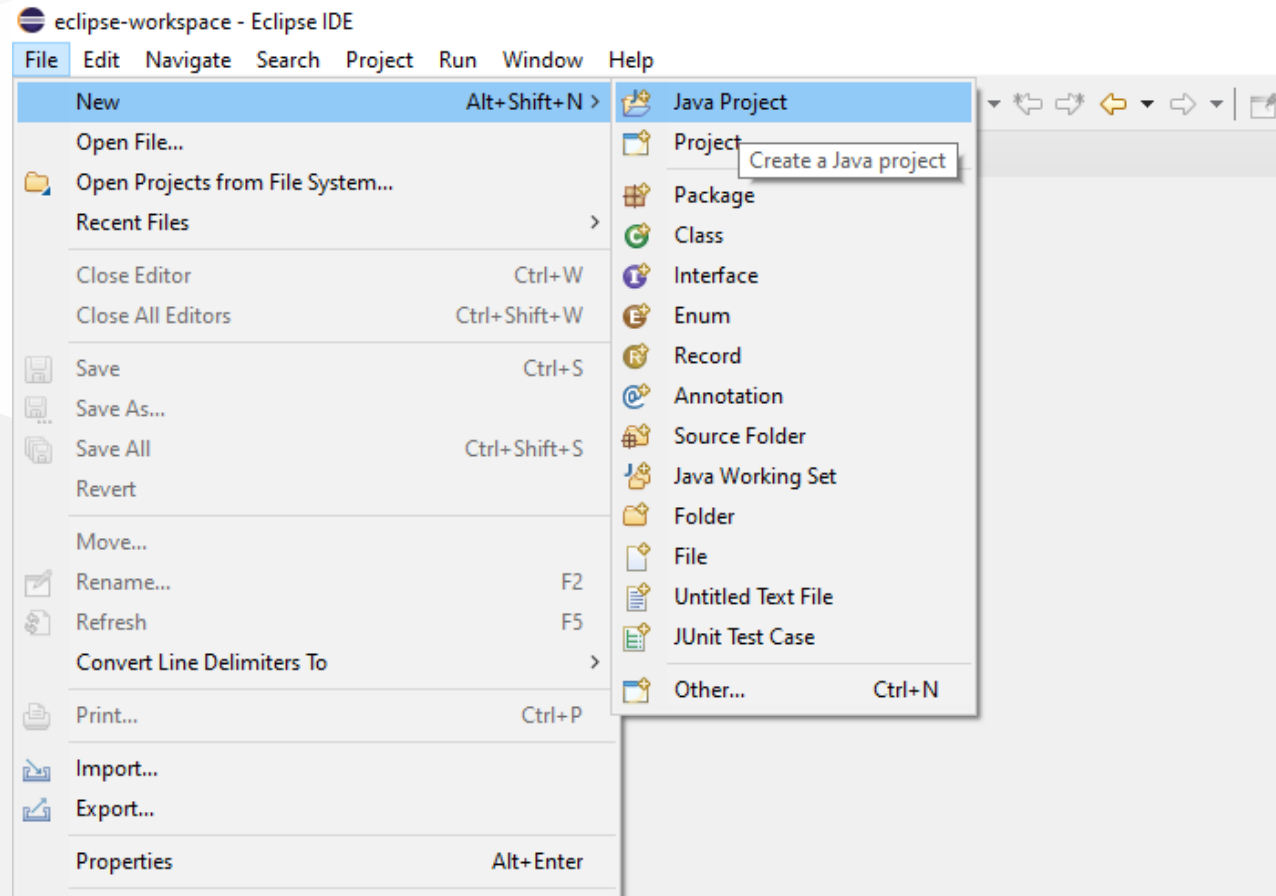
- Breakpoint: 'main' at line breakpoint NewClass.java : 20
- Current thread: 'NewClass.main:20'
- Current frame: 'main' in 'com.ucooruh.mavenproject1.NewClass'

The 'Variables' window shows the following state:

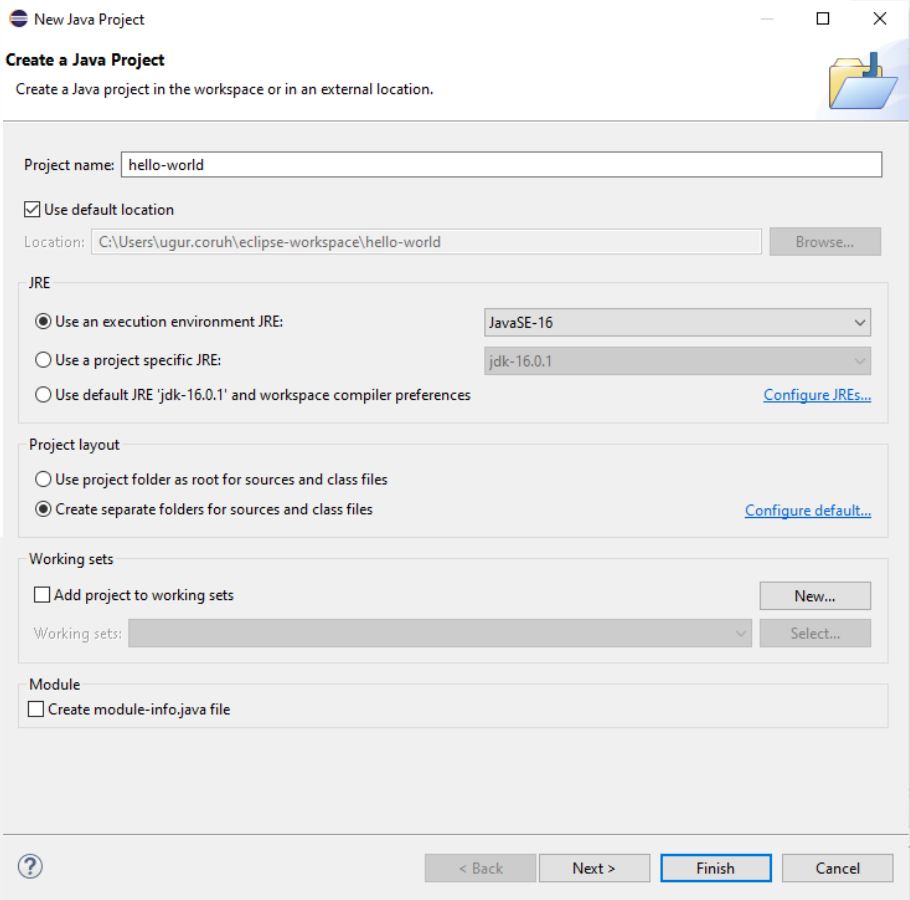
Name	Type	Value
<Enter new watch>		
Static		
args	String[]	#63(length=0)

## Eclipse (Java) (1)

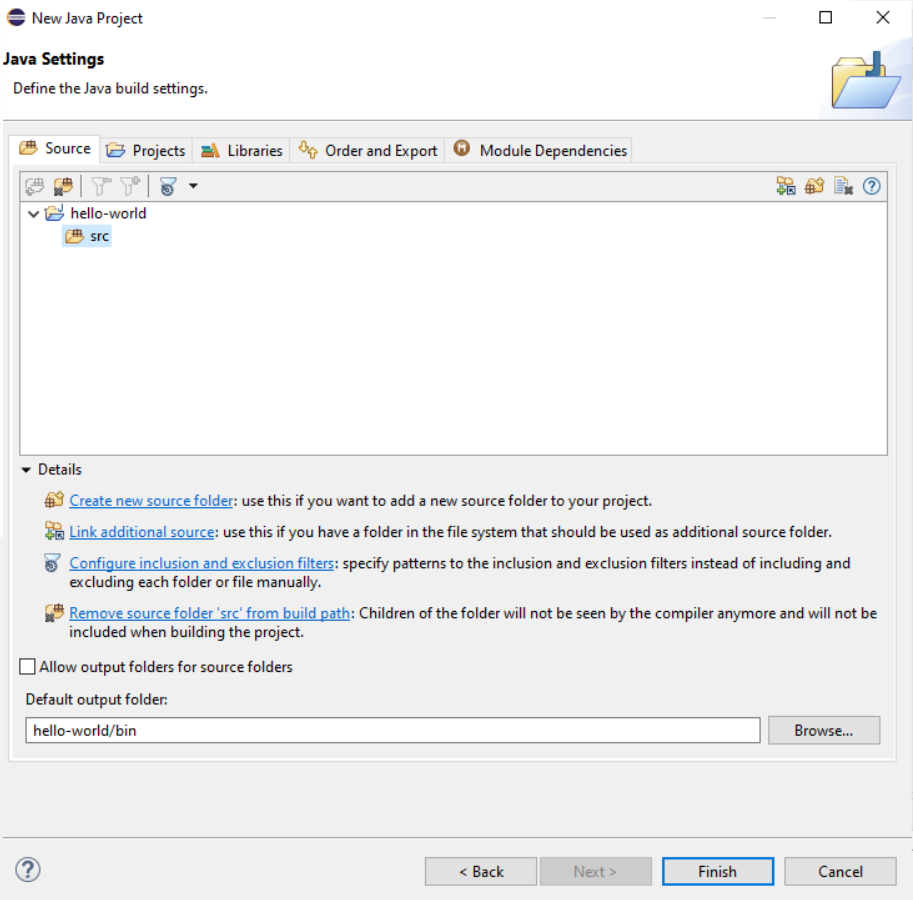
- Select File -> New Project



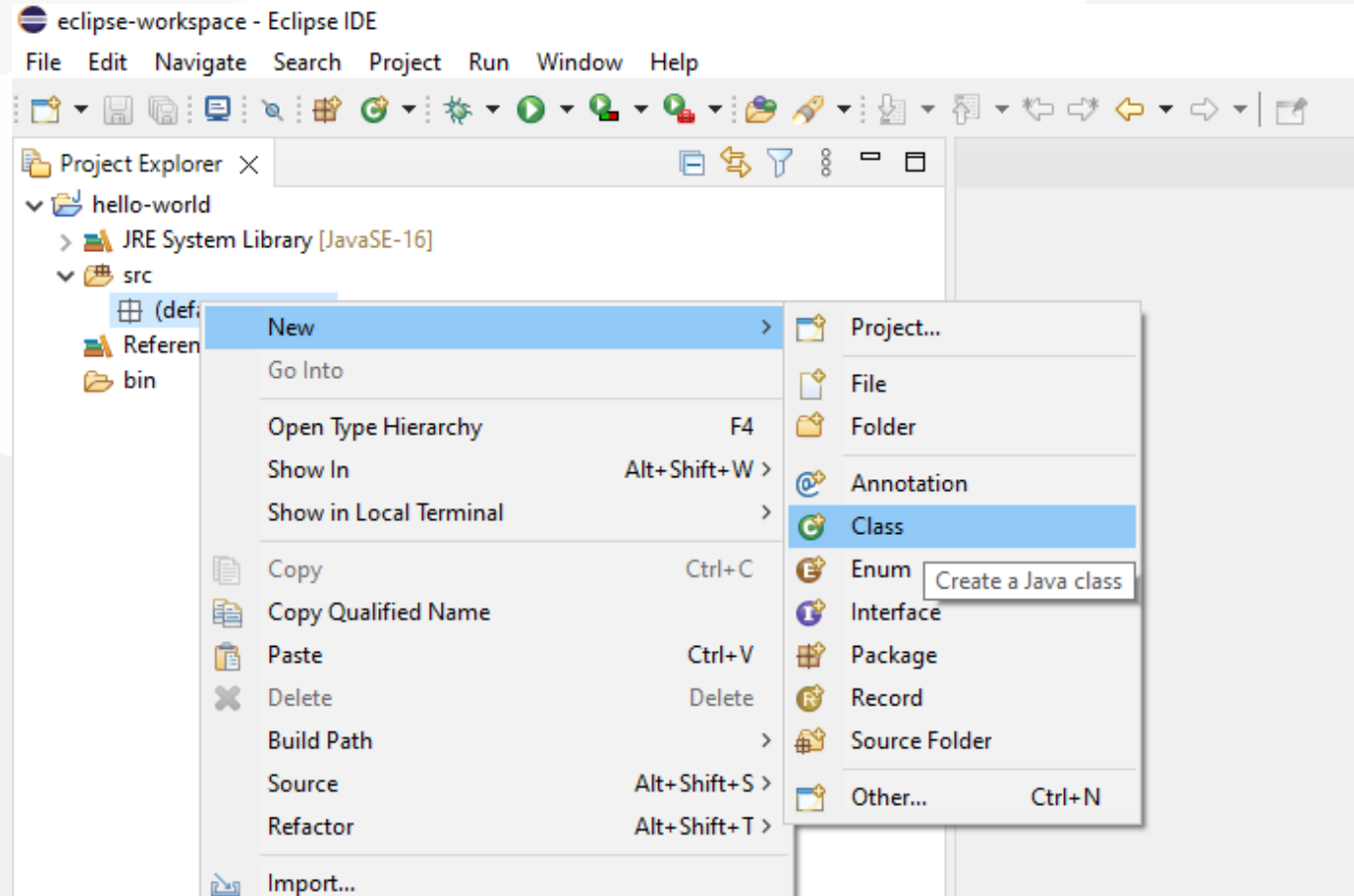
# Eclipse (Java) (2)



# Eclipse (Java) (3)



## Eclipse (Java) (4)





## Eclipse (Java) (5)

**New Java Class**

**Java Class**  
Create a new Java class.

Source folder:

Package:

Enclosing type:

Name:

Modifiers:  public  package  private  protected  
 abstract  final  static

Superclass:

Interfaces:

Which method stubs would you like to create?

public static void main(String[] args)  
 Constructors from superclass  
 Inherited abstract methods

Do you want to add comments? (Configure templates and default value [here](#))  
 Generate comments

## Eclipse (Java) (6)

- Update source

```
package ucoruh;

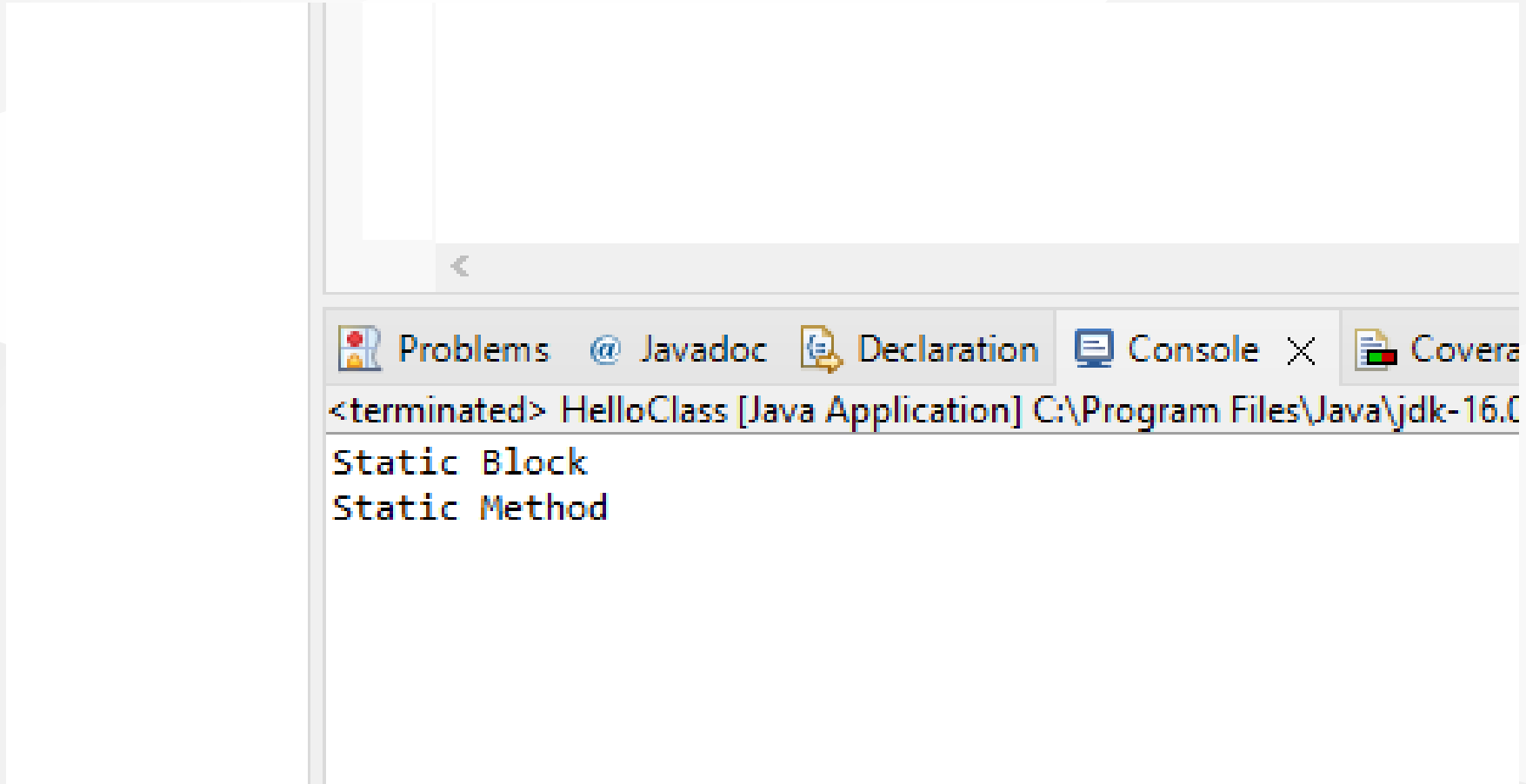
public class HelloClass {

    static {
        System.out.println("Static Block");
    }

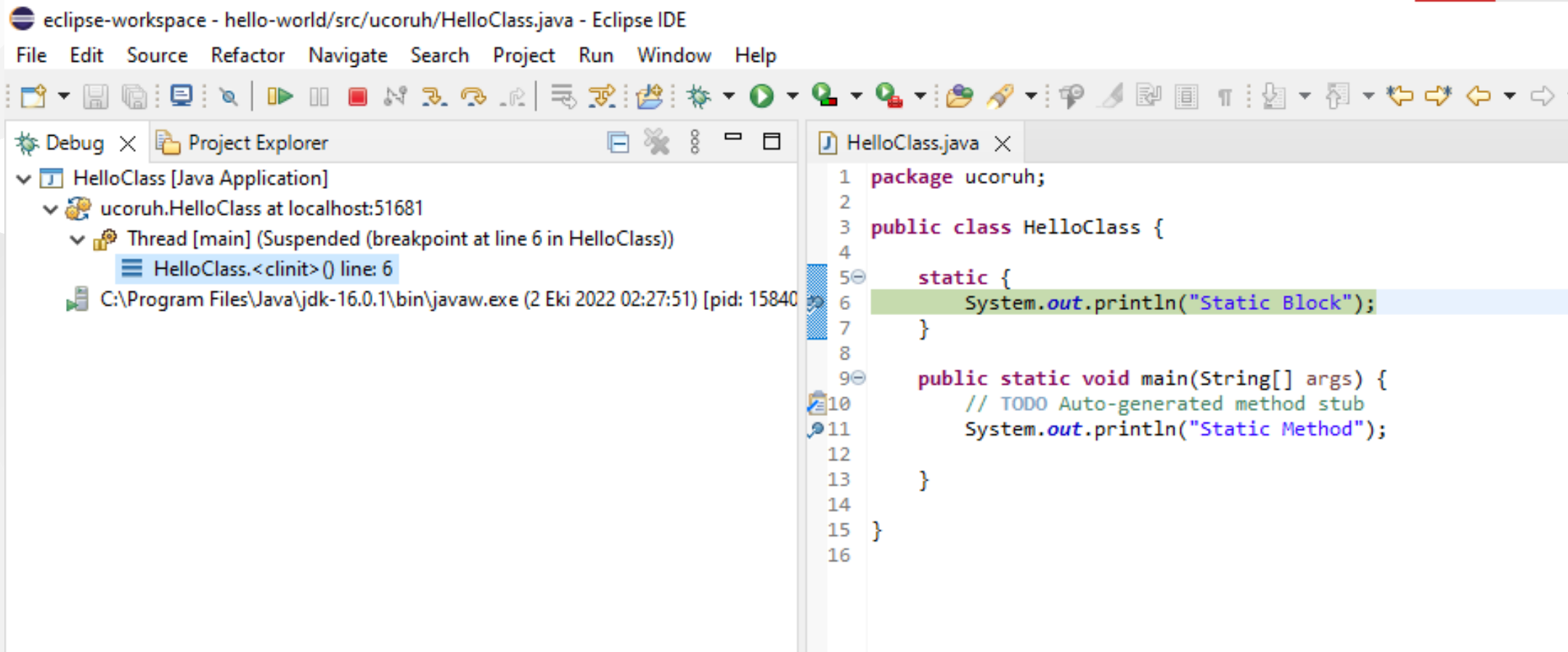
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        System.out.println("Static Method");
    }

}
```

## Eclipse (Java) (7)



## Eclipse (Java) (8)

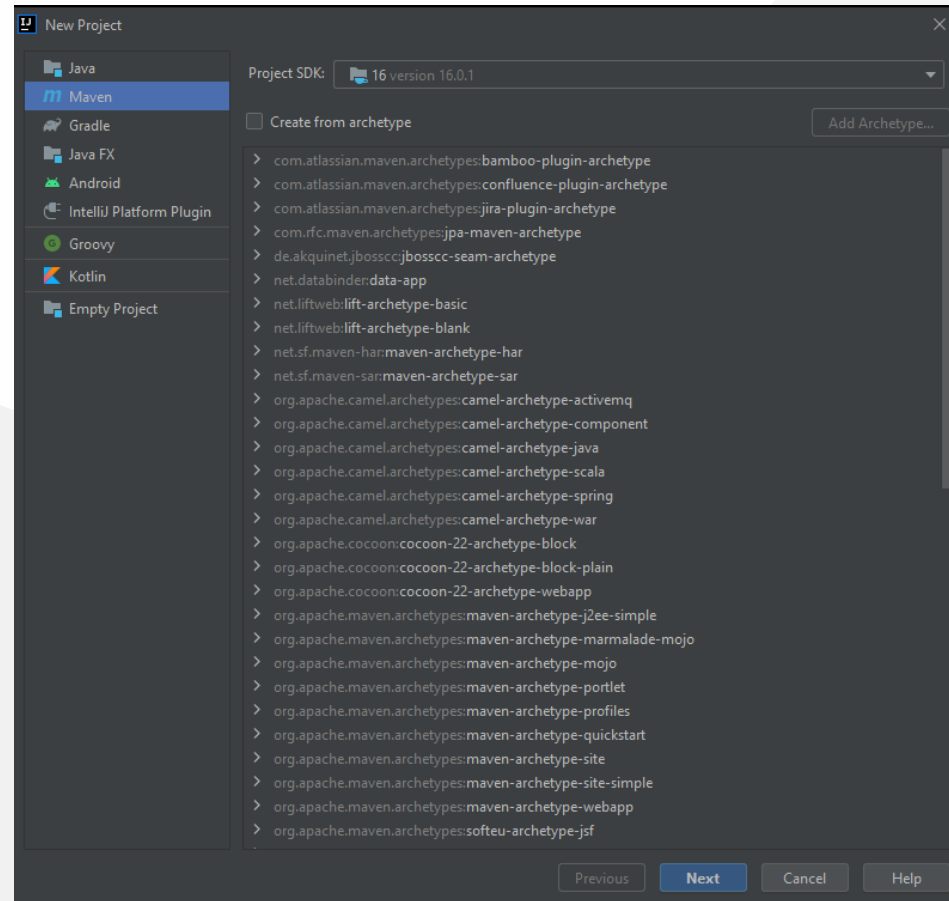


The screenshot shows the Eclipse IDE interface. The title bar reads "eclipse-workspace - hello-world/src/ucoruh/HelloClass.java - Eclipse IDE". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar contains various icons for file operations, search, and execution. The Project Explorer on the left shows the project structure: "HelloClass [Java Application]" containing "ucoruh.HelloClass at localhost:51681", which includes a "Thread [main] (Suspended (breakpoint at line 6 in HelloClass))" and "HelloClass.<clinit>() line: 6". The main editor window displays the source code for "HelloClass.java" with the following content:

```
1 package ucoruh;
2
3 public class HelloClass {
4
5     static {
6         System.out.println("Static Block");
7     }
8
9     public static void main(String[] args) {
10        // TODO Auto-generated method stub
11        System.out.println("Static Method");
12    }
13
14 }
15 }
16
```

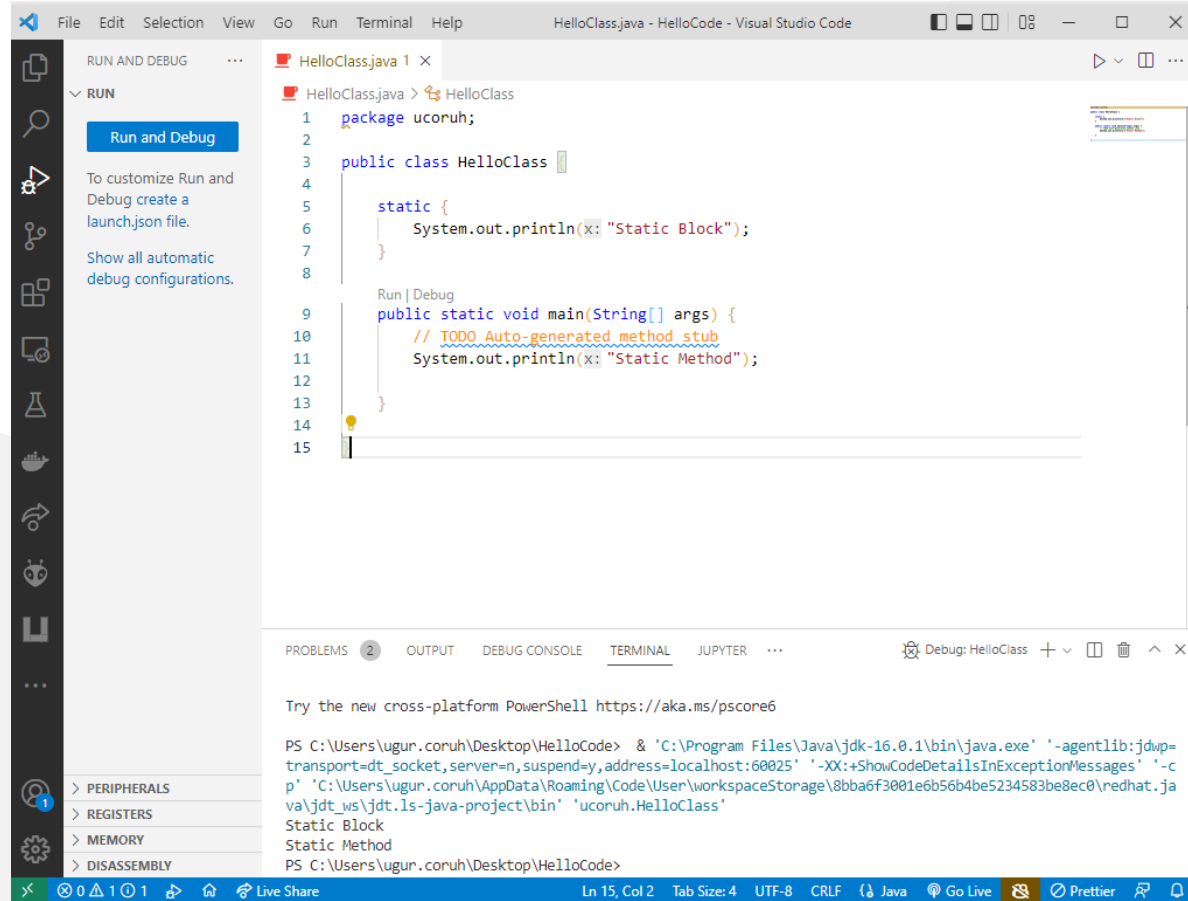
## IntelliJ Idea (Jet Brains) (Java)

- Download IntelliJ IDEA: The Capable & Ergonomic Java IDE by JetBrains
  - Select Community Version or Student Ultimate Version



## VSCoDe (Java)

- Java Extension Run&Debug Java Files



```
File Edit Selection View Go Run Terminal Help HelloClass.java - HelloCode - Visual Studio Code
HelloClass.java 1 x
HelloClass.java > HelloClass
1 package ucoruh;
2
3 public class HelloClass {
4
5     static {
6         System.out.println(x: "Static Block");
7     }
8
9     Run | Debug
10    public static void main(String[] args) {
11        // TODO: Auto-generated method stub
12        System.out.println(x: "Static Method");
13    }
14
15
PROBLEMS 2 OUTPUT DEBUG CONSOLE TERMINAL JUPYTER ...
Debug: HelloClass + - X
Try the new cross-platform PowerShell https://aka.ms/pscore6
PS C:\Users\ugur.coruh\Desktop\HelloCode> & 'C:\Program Files\Java\jdk-16.0.1\bin\java.exe' '-agentlib:jdwp=transport=dt_socket,server=n,suspend=y,address=localhost:60025' '-XX:+ShowCodeDetailsInExceptionMessages' '-cp' 'C:\Users\ugur.coruh\AppData\Roaming\Code\User\workspaceStorage\8bba6f3001e6b56b4be5234583be8ec0\redhat.ja va\jdt_ws\jdt.ls-java-project\bin' 'ucoruh.HelloClass'
Static Block
Static Method
PS C:\Users\ugur.coruh\Desktop\HelloCode>
```

## Notepad++ (Java)

- How to Compile and Run Java Programs Using Notepad++

## Cmake (Java)

- [UseJava — CMake 3.24.2 Documentation](#)
- [GitHub - ptitpoulpe/cmake-swig-java-example: An example of combining cmake, swig and java](#)



# C# Environment and Development

## Visual Studio Community Edition (C#)

```
//TODO//
```



## Notepad++ (C#)

- This use command-line utilities for csharp, nppexec should be configured for this utility.
- [Compiling/Executing a C# Source File in Command Prompt - Stack Overflow](#)

```
c:\windows\Microsoft.NET\Framework\v3.5\
```

```
c:\windows\Microsoft.NET\Framework\v3.5\bin\csc.exe  
    /t:exe /out:MyApplication.exe MyApplication.cs ...
```

## Cmake (C#)

- [GitHub - crud89/DotNetWithCMake: Your swiss army knife for creating .NET assemblies with CMake and integrating unmanaged code.](#)

## Common Tools and Platforms

## Fatih Kalem



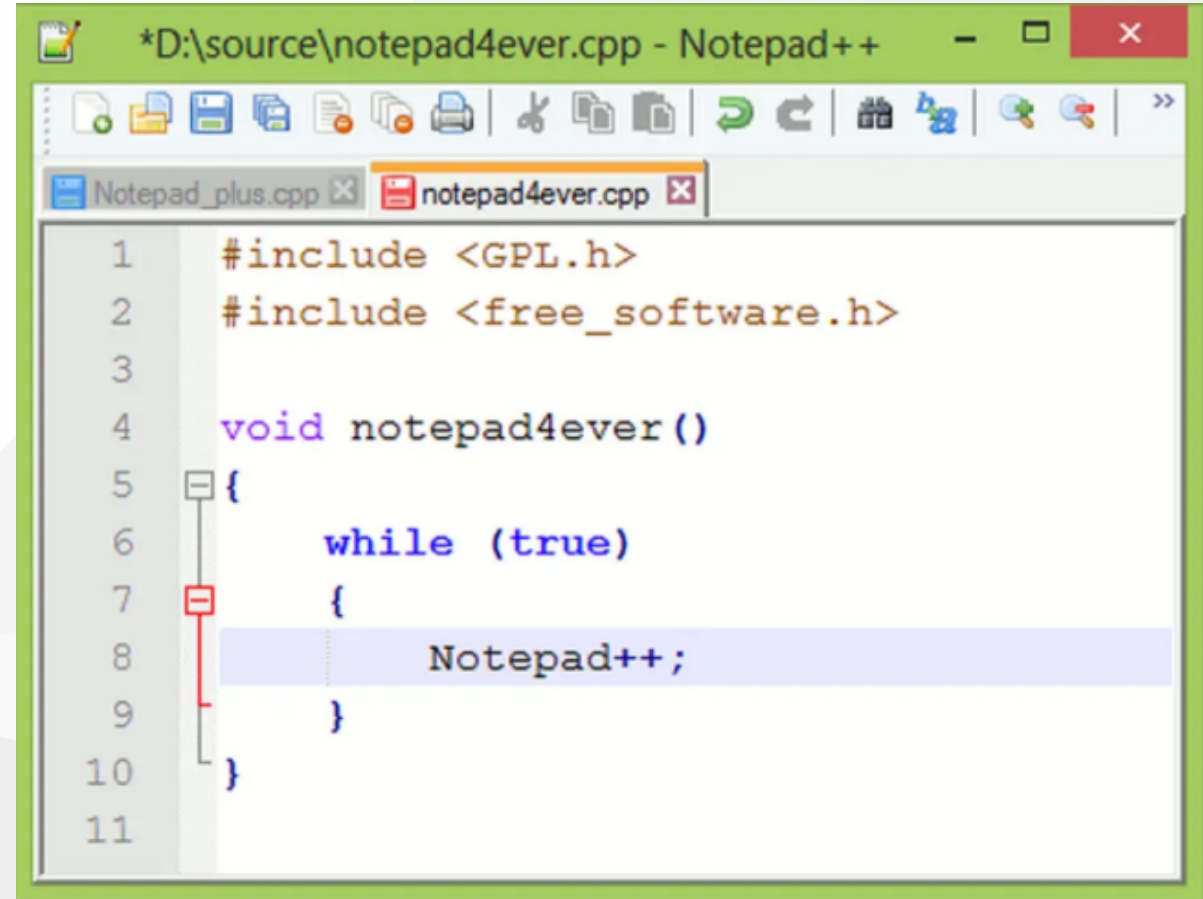
[https://cdnvideo.eba.gov.tr/fatihkalem/fatihkalem\\_portable.zip](https://cdnvideo.eba.gov.tr/fatihkalem/fatihkalem_portable.zip)

[https://cdnvideo.eba.gov.tr/fatihkalem/fatihkalem\\_setup.exe](https://cdnvideo.eba.gov.tr/fatihkalem/fatihkalem_setup.exe)



## Notepad++ (Notepad for Source Code)

[Downloads | Notepad++](#)



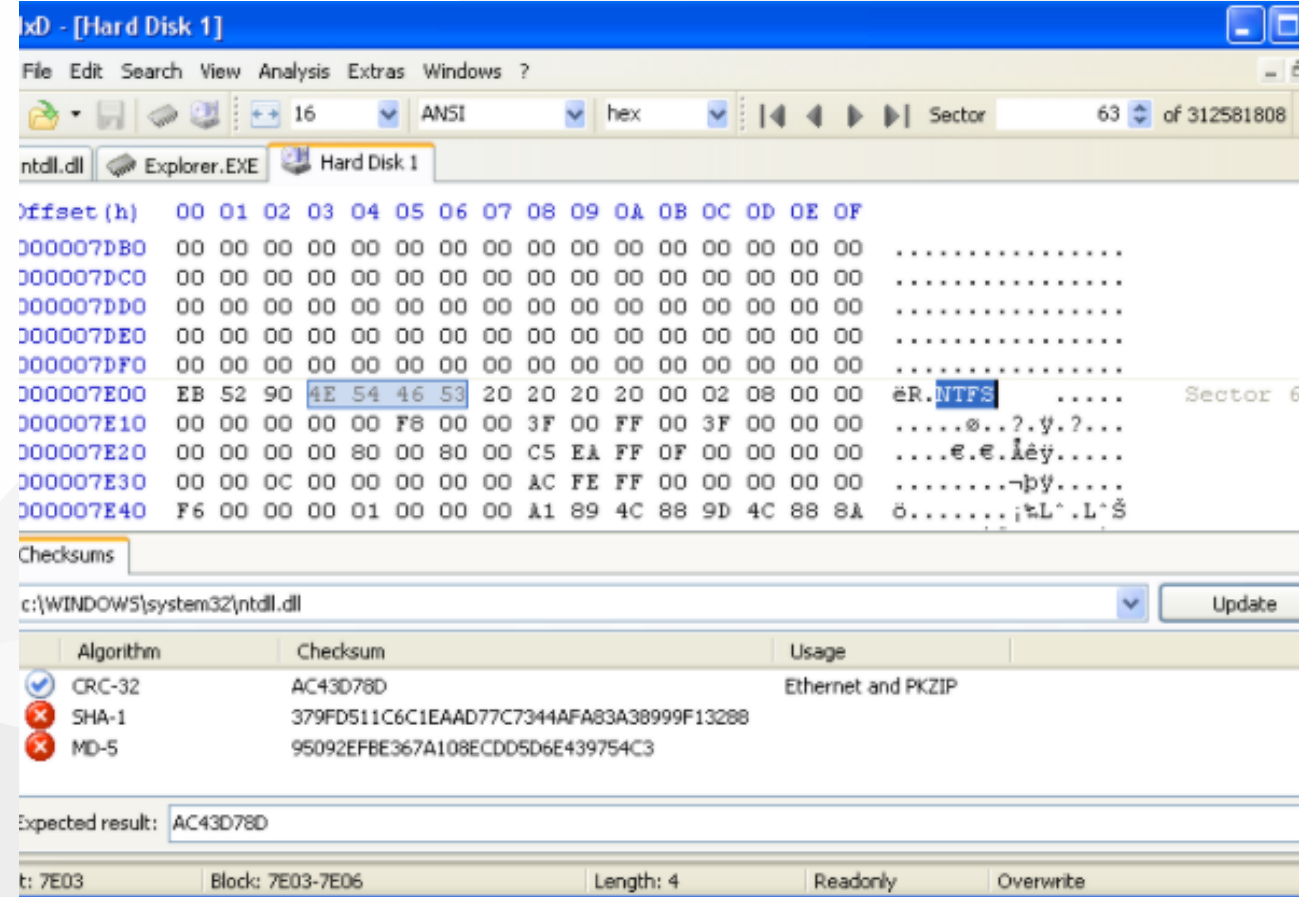
```
*D:\source\notepad4ever.cpp - Notepad++
#include <GPL.h>
#include <free_software.h>

void notepad4ever()
{
    while (true)
    {
        Notepad++;
    }
}
```

## HxD (Hex Editor)



HxD - Freeware Hex Editor and Disk Editor |  
[mh-nexus](http://mh-nexus.com)

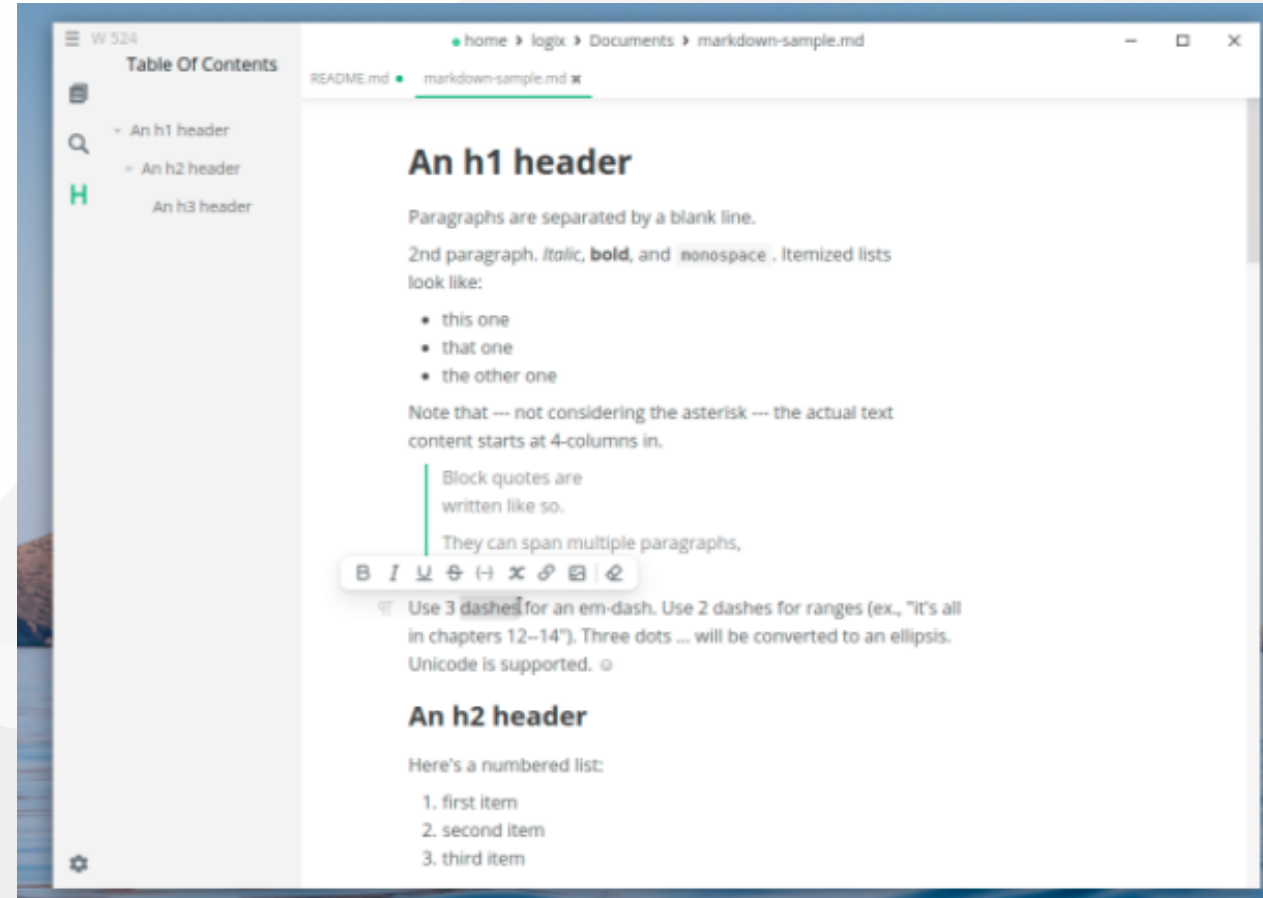




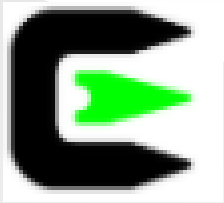
## MarktextApp (Markdown Syntax Editor)



- <https://marktext.app/>
- <https://github.com/marktext/marktext/releases>
- Download latest version
  - <https://github.com/marktext/marktext/releases/tag/v0.17.1>



## Cygwin (Linux environment for Windows)



- <https://www.cygwin.com/>

```

Cygwin Terminal Output:
bin cygdrive cygwin.bat cygwin.ico etc
$ cd cygdrive/
$ ls
$ cd c
$ ls
AUTORUN.INI      MSOCache
AUTOEXEC.BAT    My Music
BOOT.INI         MYDETECT.COM
BOOTSECT.DOS    Oracle
CONFIG.SYS       PRELOAD.AAA
CollideSoftware Program Files
Daten            RECYCLER
Download         Recycled
IO.SYS           SSSSetup.log
Images          SYSINFO
Inetpub         System Volume Information
Kunden.mp3      TerraTec
Lotus           Tomcat 5.0
MSDOS.SYS       URLCACHE
WINDOWS
  
```

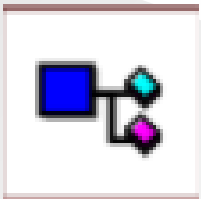
Name	Size	MTIME	Name	Size	MTIME
./JREx	0	Jun 5 10:51	/bin		
./barland	0	Jun 10 01:53	/cygdrive		Jun 13 22:34
./jbuilder2005	0	Jun 10 02:02	/etc		Jan 1 1970
./nc	0	Jun 13 22:30	/lib		Jun 13 22:34
./netbeans	0	Jun 10 02:05	/tmp		Jun 13 22:04
./primitivc2005	0	Jun 10 02:02	/usr		Jun 13 22:30
./qualitycentral	0	Jun 10 01:54	/var		Jun 13 22:34
./AnagramGame	0	Jun 10 02:22	*/cygwin.bat	57	Jun 13 22:36
./Anwendungsdaten	0	Jun 12 20:20	*/cygwin.ico	7022	Jun 13 22:36
./Application Data	0	Mar 22 18:10	*/null	0	Jun 13 22:03
./Cookies	0	Jun 7 23:48			
./Desktop	0	Jun 13 22:50			
./Druckumgebung	0	Sep 14 2004			
./Eigene Dateien	0	May 5 23:45			
./Favoriten	0	May 13 14:35			
./..			/bin		

```

PuTTY Terminal Output:
login as: root
Password:
Last login: Mon Jun 13 22:39:05 2005 from graceyves.qy8128
Have a lot of fun...
graceyves:~ # kwin &
[1] 7644
graceyves:~ # kbldsysoccs running...
Launched ok, pid = 7660
xclock &
[2] 7661
graceyves:~ #
  
```

# Dependency Walker (32-bit or 64-bit Windows module dependency checker)



- <https://www.dependencywalker.com/>

The screenshot shows the Dependency Walker interface for the file STOOGES.EXE. The left pane displays a dependency tree where STOOGES.EXE depends on LARRY.DLL, KERNEL32.DLL, NTDLL.DLL, and CURLY.DLL. CURLY.DLL further depends on SHEMP.DLL and MOE.DLL. MOE.DLL depends on KERNEL32.DLL and NTDLL.DLL.

The right pane shows a list of functions and their entry points for the selected module (CURLY.DLL):

PI^	Ordinal	Hint	Function	Entry Point
■	N/A	N/A	IsKnucklehead	Not Bound
■	N/A	N/A	int SaySoitenly(char *,...)	Not Bound
!!!				
E^	Ordinal	Hint	Function	Entry Point
■	4 (0x0004)	1 (0x0001)	int SaySoitenly(char *,...)	SHEMP.?SaySoitenly@YAHP/
■	5 (0x0005)	2 (0x0002)	DoinkLarrysEye	0x00001010
■	3 (0x0003)	0 (0x0000)	void SayPoifect(_int64)	0x00001020
■	1 (0x0001)	N/A	N/A	0x00001020
■	2 (0x0002)	3 (0x0003)	DoinkMoesEye	SHEMP.DoinkMoesEye
!!!				

The bottom pane shows a list of loaded modules with their file and link time stamps, file sizes, attributes, and checksums:

Module ^	File Time Stamp	Link Time Stamp	File Size	Attr.	Link Checksum	Real Checksum	CPU	Subsyste
CURLY.DLL	11/14/2006 5:17p	11/14/2006 5:13p	2,560	A	0x0000F739	0x0000F759	x86	GUI
KERNEL32.DLL	08/30/2006 1:22a	08/30/2006 1:20a	871,424	A	0x000E388E	0x000E388E	x86	Console
LARRY.DLL	11/14/2006 5:13p	11/14/2006 5:13p	2,560	A	0x000053DB	0x000053DB	x86	GUI
MOE.DLL	11/14/2006 5:15p	11/14/2006 5:15p	2,560	A	0x0000B191	0x0000B191	x86	GUI
NTDLL.DLL	08/30/2006 1:23a	08/30/2006 1:21a	1,147,664	A	0x00125FA5	0x00125FA5	x86	Console
SHEMP.DLL	11/14/2006 5:13p	11/14/2006 5:13p	2,560	A	0x00001CE7	0x00001CE7	x86	GUI

The bottom-most pane shows a log of system events:

```

00:00:00.093: LoadLibraryA("Moe.dll") called from "STOOGES.EXE" at address 0x00401024 by thread 1.
00:00:00.093: Loaded "MOE.DLL" at address 0x00020000 by thread 1. Successfully hooked module.
00:00:00.093: DllMain(0x00020000, DLL_PROCESS_ATTACH, 0x00000000) in "MOE.DLL" called by thread 1.
00:00:00.093: DllMain(0x00020000, DLL_PROCESS_ATTACH, 0x00000000) in "MOE.DLL" returned 1 (0x1) by thread 1.
00:00:00.093: LoadLibraryA("Moe.dll") returned 0x00020000 by thread 1.
00:00:00.109: GetProcAddress(0x00020000 [MOE.DLL], "SmackCurly") called from "STOOGES.EXE" at address 0x0040102B and returne
    
```

## Doxygen (Code Documentation)



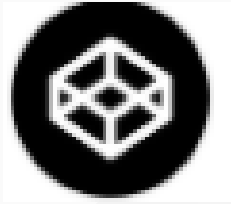
Doxygen: Doxygen

A screenshot of a Doxygen-generated web page for the "ME Project 1.0". The page is titled "Classes" and shows a navigation menu on the left with "ACMESmartphone" selected. The main content area displays the "acme.ACMESmartphone Class Reference". Under "Public Member Functions", there are two entries: "findRoadRunner (String city, String state) throws IOException" and "zapRoadRunner (int voltage) throws IOException". Under "Public Attributes", there is one entry: "LongLat = \"Longitude = 39.2334, Latitude = 41.4899\"". A "Detailed Description" section follows, stating "Works like a regular smartphone but also tracks roadrunners." and "The ACME Smartphone can perform similar functions as other smartphones, such as making phone calls, se...". The footer of the page indicates it was generated on Sun Sep 27 2015 09:05:27 for the ACME Project by Doxygen.

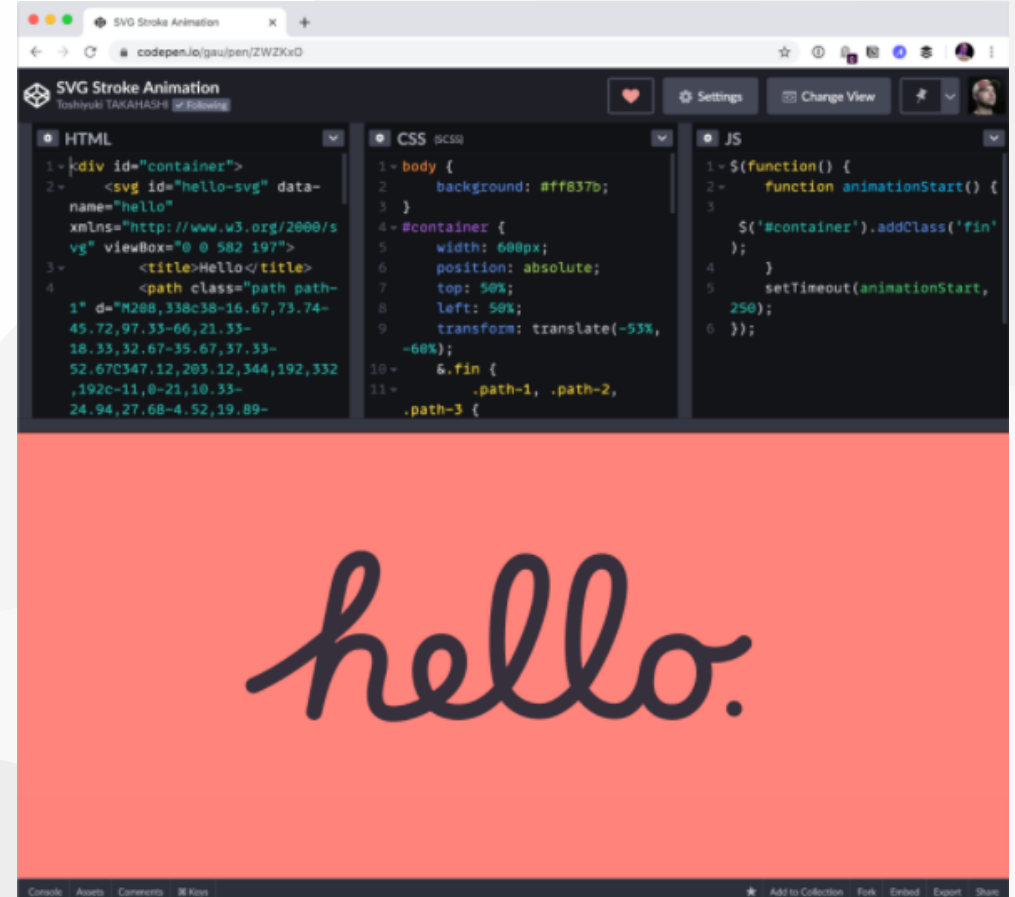
## Sonarlint (Code Quality and Code Security Extension)



<https://www.sonarlint.org/>



- <https://codepen.io/>
- CodePen is a social development environment. At its heart, it allows you to write code in the browser, and see the results of it as you build.
- A useful and liberating online code editor for developers of any skill, and particularly empowering for people learning to code. We focus primarily on front-end languages like HTML, CSS, JavaScript, and preprocessing syntaxes that turn into those things



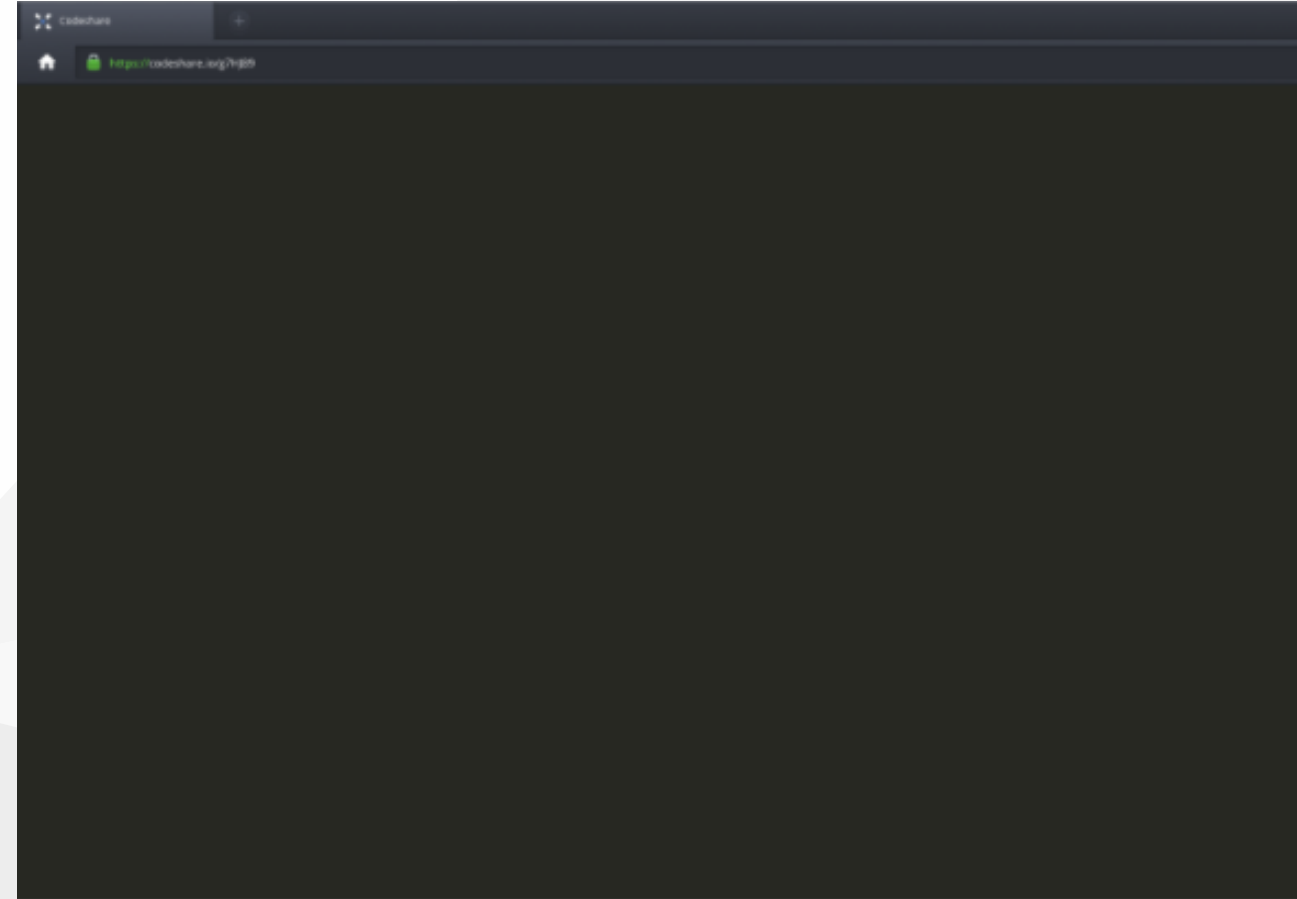
## Codepen.io (online code sharing)

- Credit Card Sample on Codepen
  - <https://codepen.io/quinlo/pen/YONMEa>
- Checkout trends <https://codepen.io/trending>

## Codeshare.io (real-time code sharing)

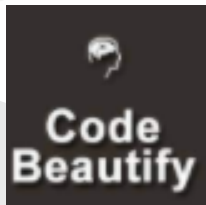


- <https://codeshare.io/>
- Share Code in Real-time with Developers, An online code editor for interviews, troubleshooting, teaching & more...





## Codebeautify.org (online data conversion tools)



- Has several tools for developers (Code Formatter, JSON Beautifier, XML Viewer, Hex Converters and more...)
- <https://codebeautify.org/>

- Asciflow provides ascii based drawings that you can copy directly to textfiles and source codes.

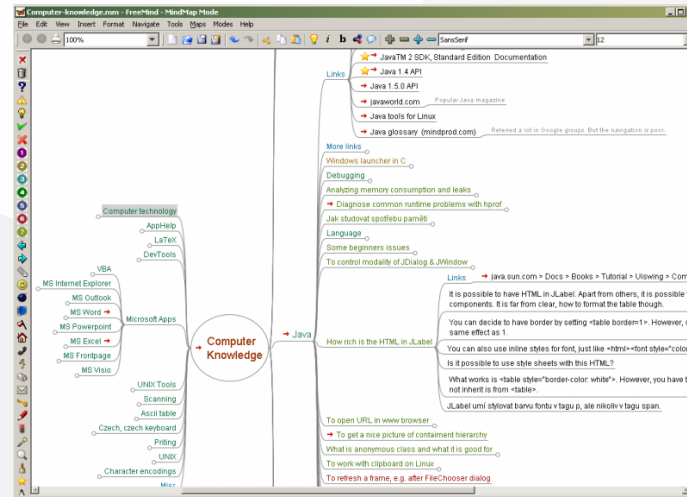
Visit the following link

- <https://asciiflow.com/>

The screenshot shows the Asciflow web application. On the left is a navigation menu with the following items: File (with download, add, and up arrows), Default drawing (263B) (with a file icon and three dots), Edit (with an up arrow), Boxes (with a square icon), Select & Move (with a cursor icon), Freeform (with a squiggly line icon and a close button 'x'), Arrow (with a zigzag arrow icon and is currently selected), Line (with a straight line icon), and Text (with a 'T' icon). The main drawing area is a grid with two empty rectangles and an ASCII art drawing of a house. The house is composed of 'x' characters forming a roof, walls, and a chimney. A blue highlight is visible on one of the 'x' characters in the house's body. To the right of the house, there is a vertical line with a horizontal arrow pointing right from its top and a horizontal arrow pointing left from its bottom.

## Freemind (opensource mindmap application)

- Freemind is open source java based desktop mindmap application. Can export files to several formats
  - [Main Page - FreeMind](#)

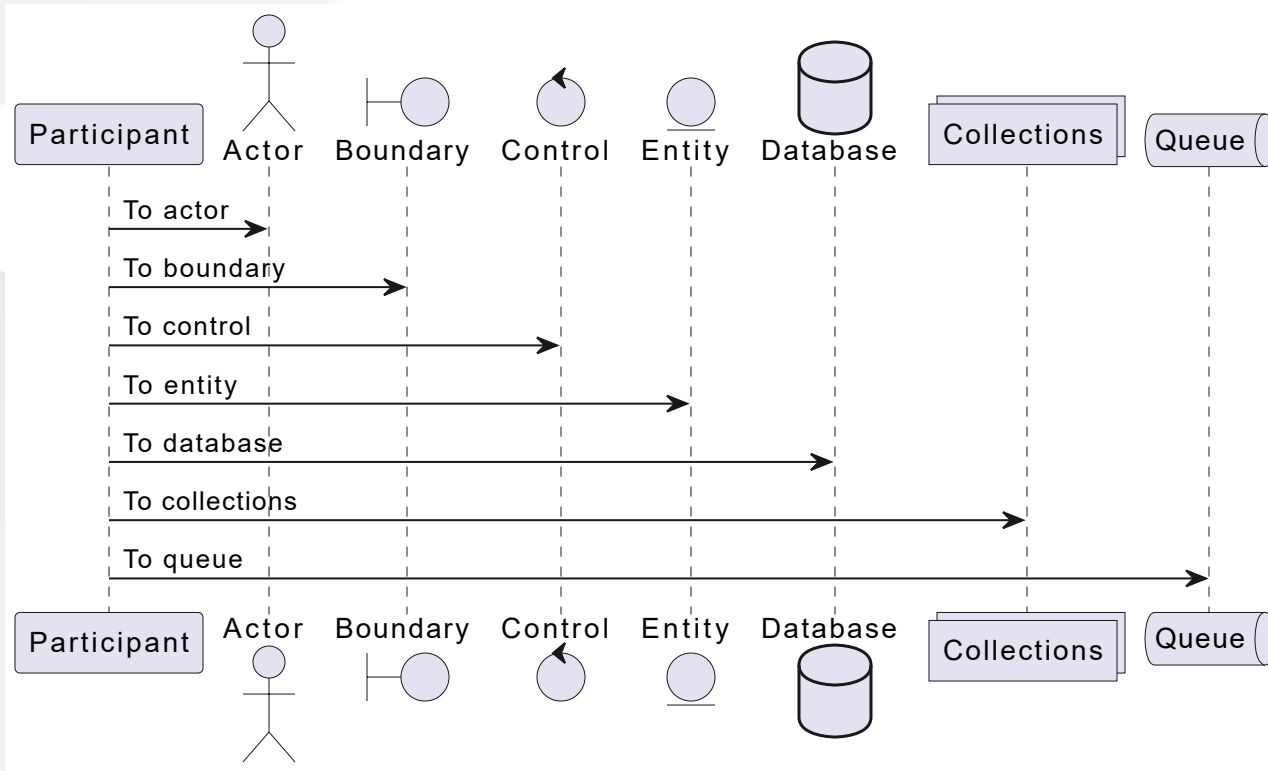


## Mockup Designers

- Mockflow
  - [Signup - MockFlow](#)
- Wireflow
  - <https://wireflow.co/>

# PlantUML (software designer)

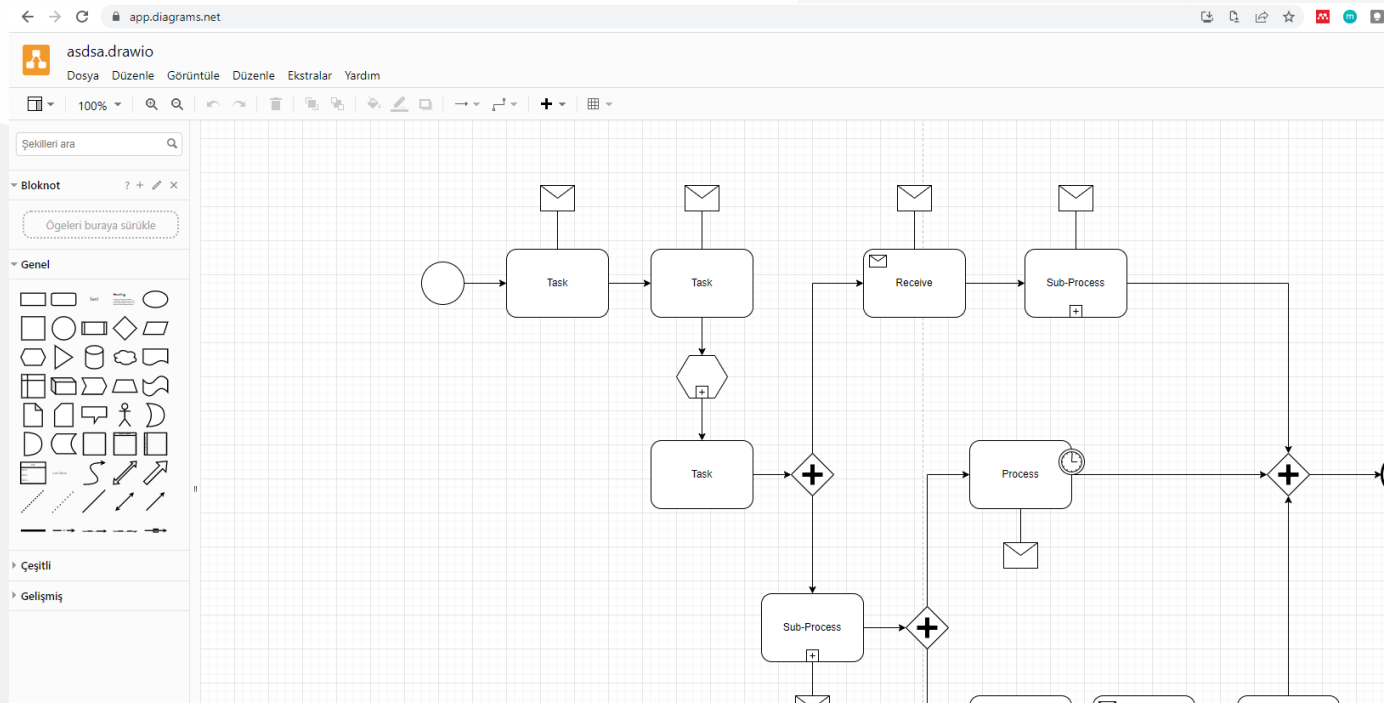
- Text based designer for software engineers
  - <https://plantuml.com/>



- Also visit course notes that related to plantuml [CE204 Object-Oriented Programming - RTEU](#)  
[C204 Object Oriented Programming Course Notes](#)

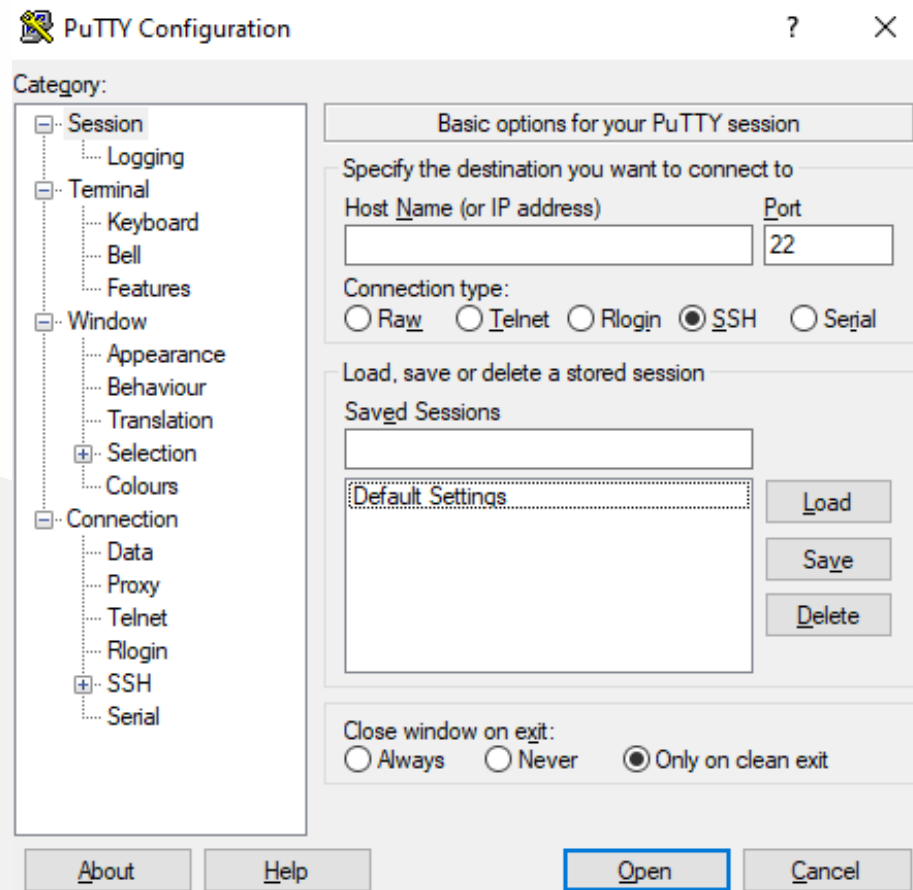
## Drawio (drawing tool)

- Online and Offline Drawing Tool
  - <https://app.diagrams.net/>
- Offline Installer
  - [Releases · jgraph/drawio-desktop · GitHub](#)

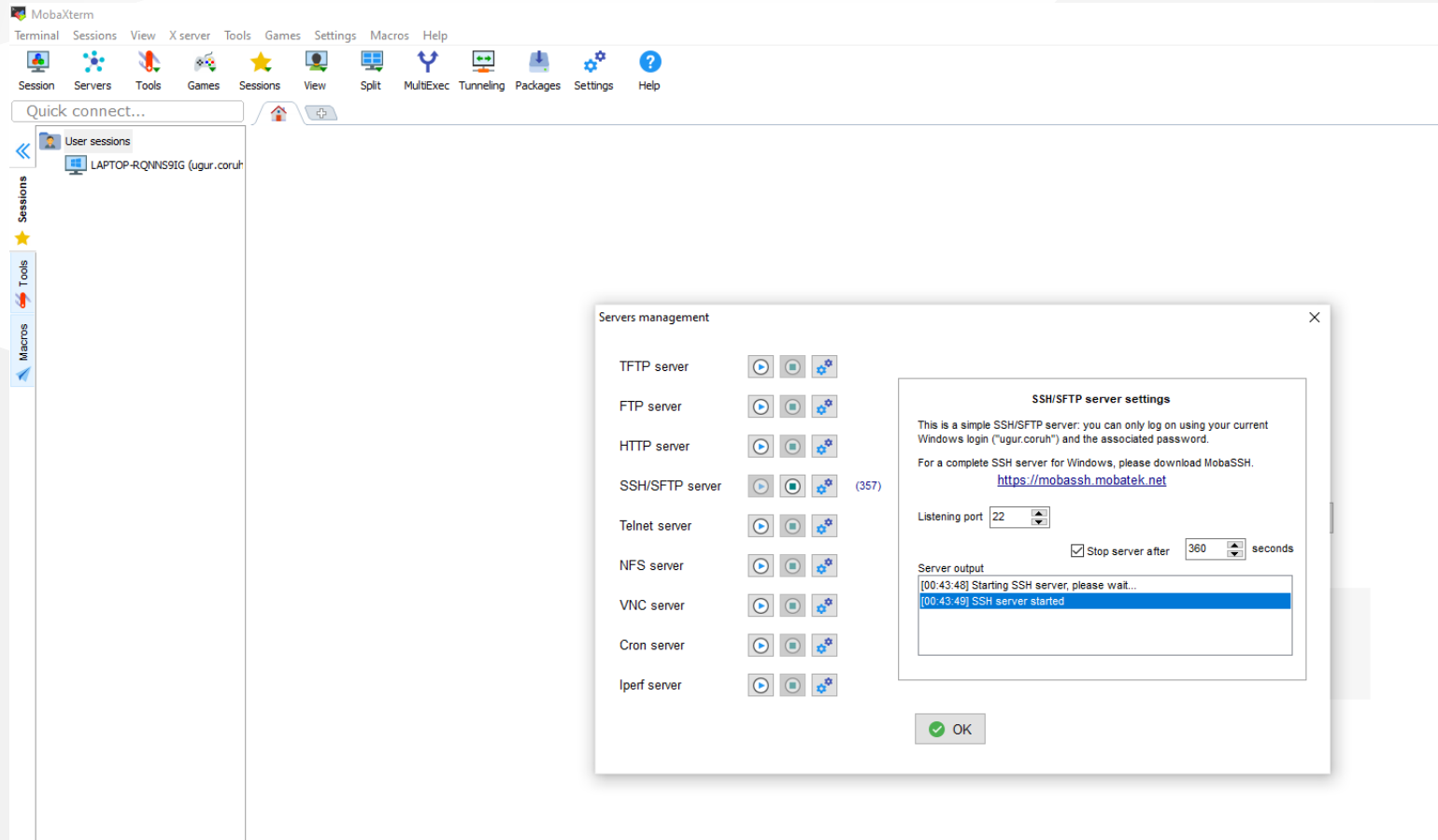


## Putty (Remote Connection)

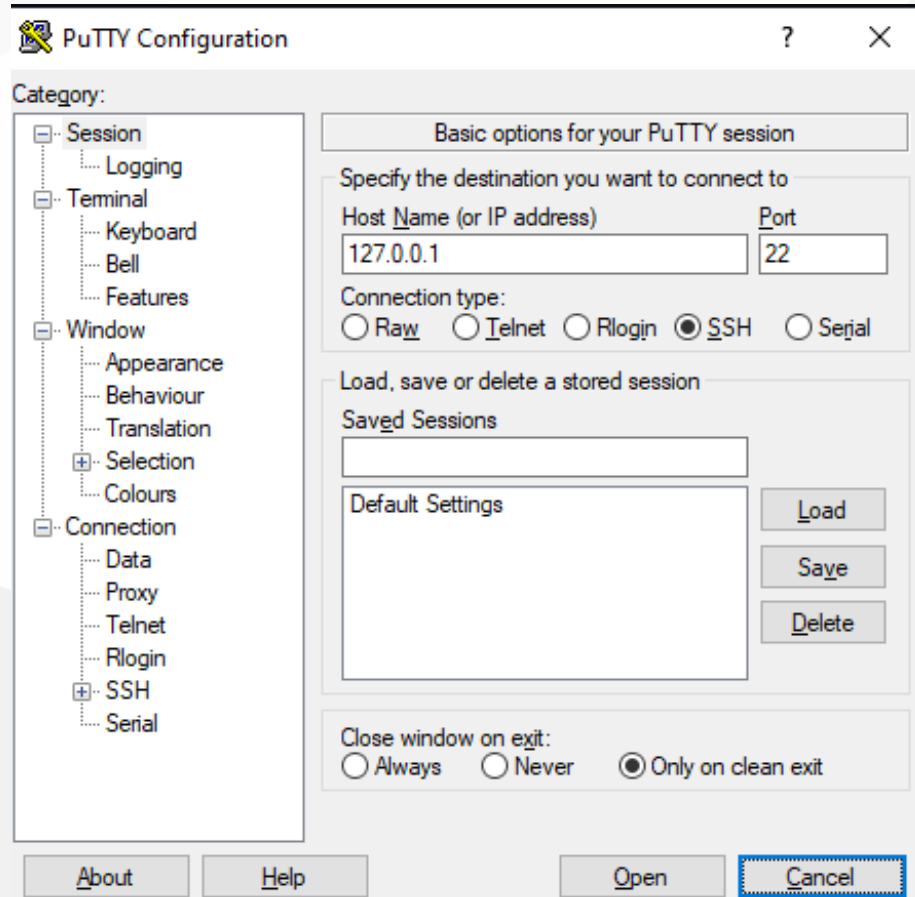
- Commonly use for SSH connection



- We can run a SSH server with MobaXterm and can connect to same computer with Putty.







```
127.0.0.1 - PuTTY
login as: ugur.coruh
ugur.coruh@127.0.0.1's password:
Last login: Wed Jun 29 22:34:28 2022 from 192.168.1.170

[2022-10-02 00:45.57] ~
[ugur.coruh.LAPTOP-RQNS9IG] b dir
BDScan
ArduinoSketchBook
Bachelor
Backups
Book Library
Cloud
Consultancy
CoruhArge
Database
Dataset Library
Desktop
Doktora
GalaxyS4
Github
Images Icons
LauncherFolder
[ugur.coruh.LAPTOP-RQNS9IG] b
```

## • How to Download and Upload Files over SSH – TecAdmin

CE103 Algorithms and Programming I

Here are some useful examples for downloading files from the remote system over SSH protocol.

- This will connect to example.com server with user “username” and copy the **/backup/file.zip** file to local system directory **/local/dir**. To use this command replace the values as per your environment.

```
scp username@example.com:/backup/file.zip /local/dir
```

- If the SSH is running on a non-standard port, You can specify the port using **-P** option with SCP command.

```
scp -P 2222 username@example.com:/backup/file.zip /local/dir
```

- If your remote server required a private key to connect server, You can use **-i** followed by a private key file path to connect your server using the SCP command. This can be helpful for AWS servers.

## Upload file using SSH

You can also upload files to the remote server using SSH protocol using the SCP command. Use the following example command for uploading files to the SSH server.

```
scp file.zip username@example.com:/remote/dir
```

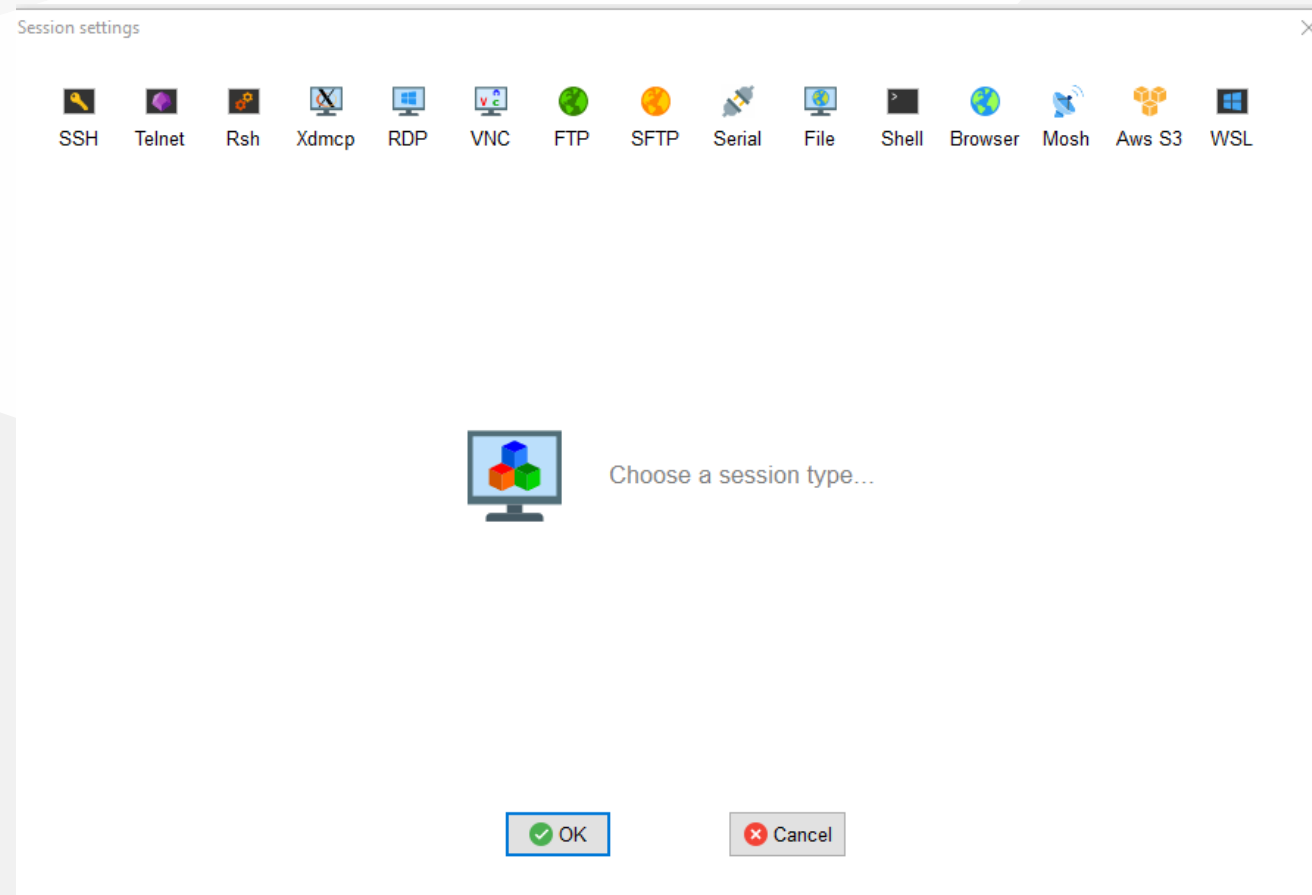
Similarly you can use **-P** switch to define port of the SSH server and **-i** to define private key for the user authentication.

- Also you can use SSH tunnels for remote code development
  - Developing on Remote Machines using SSH and Visual Studio Code
  - Visual Studio Code Server




























## MobaXterm (Remote Connection)


- Multipurpose Remote Connection Toolkit



Servers management ✕

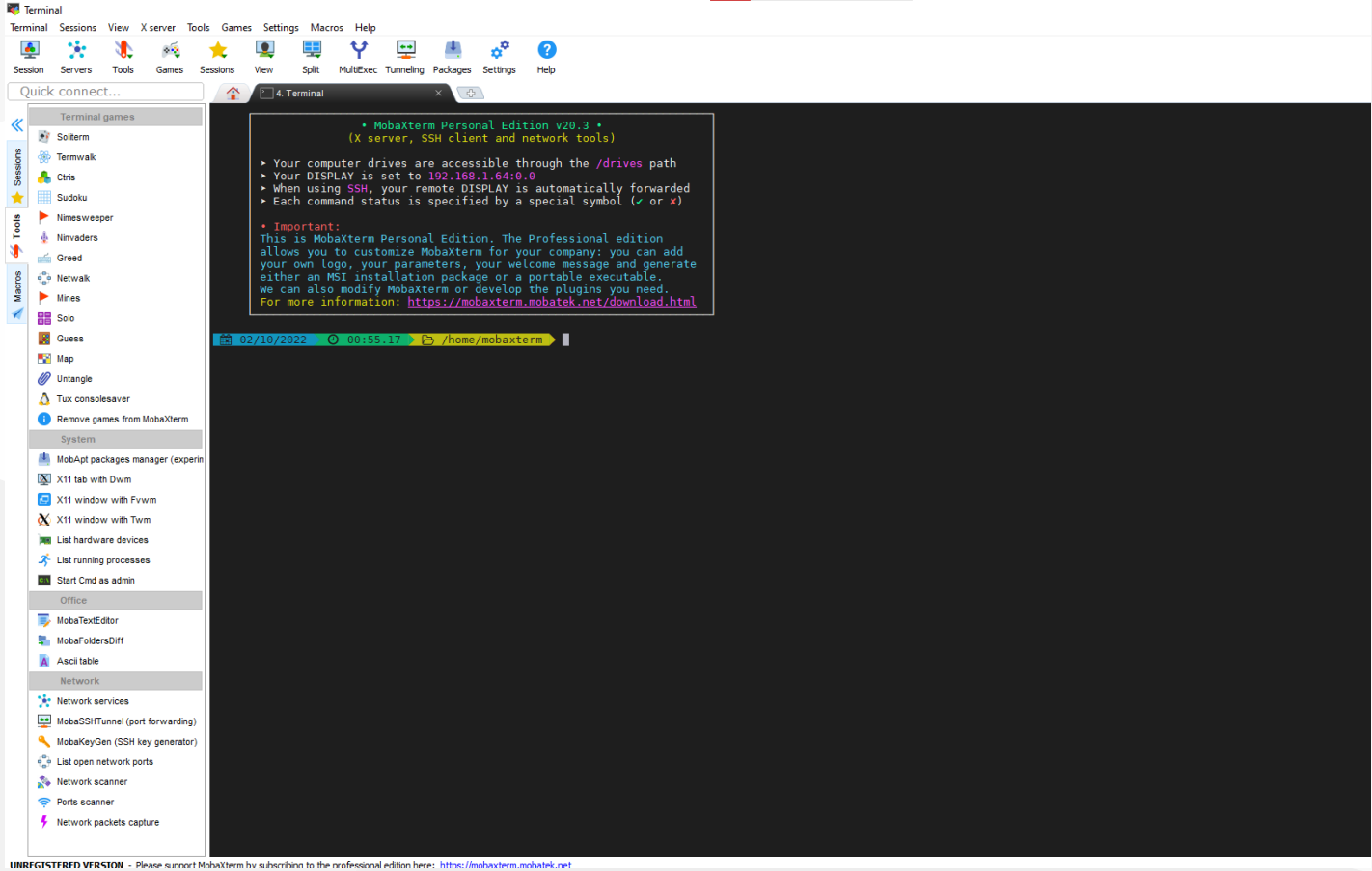
TFTP server	  
FTP server	  
HTTP server	  
SSH/SFTP server	  
Telnet server	  
NFS server	  
VNC server	  
Cron server	  
lperf server	  

Welcome to  
MobaXterm Network Services



✔ OK

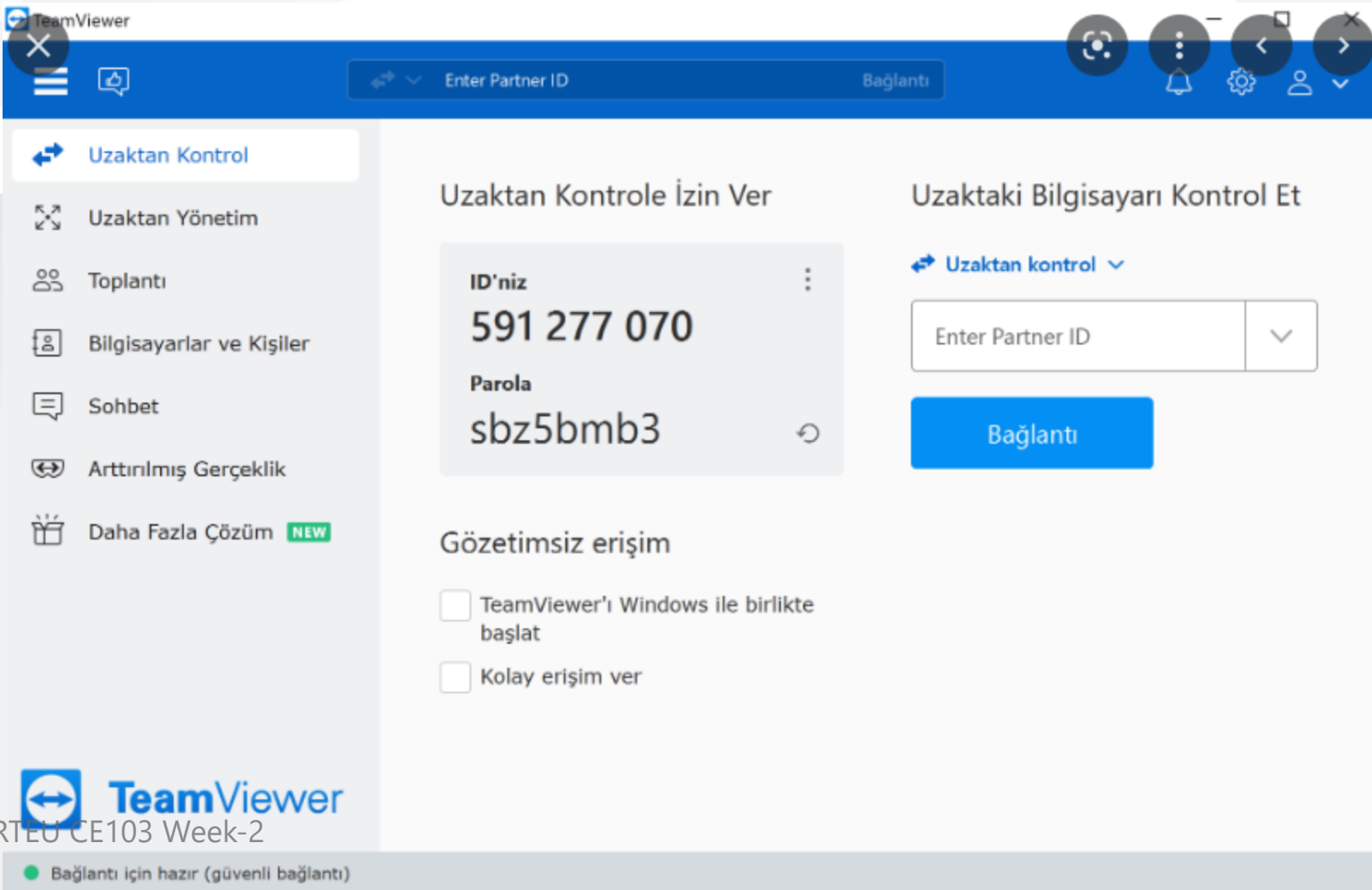




# Teamviewer (Remote Connection)

CE103 Algorithms and Programming I

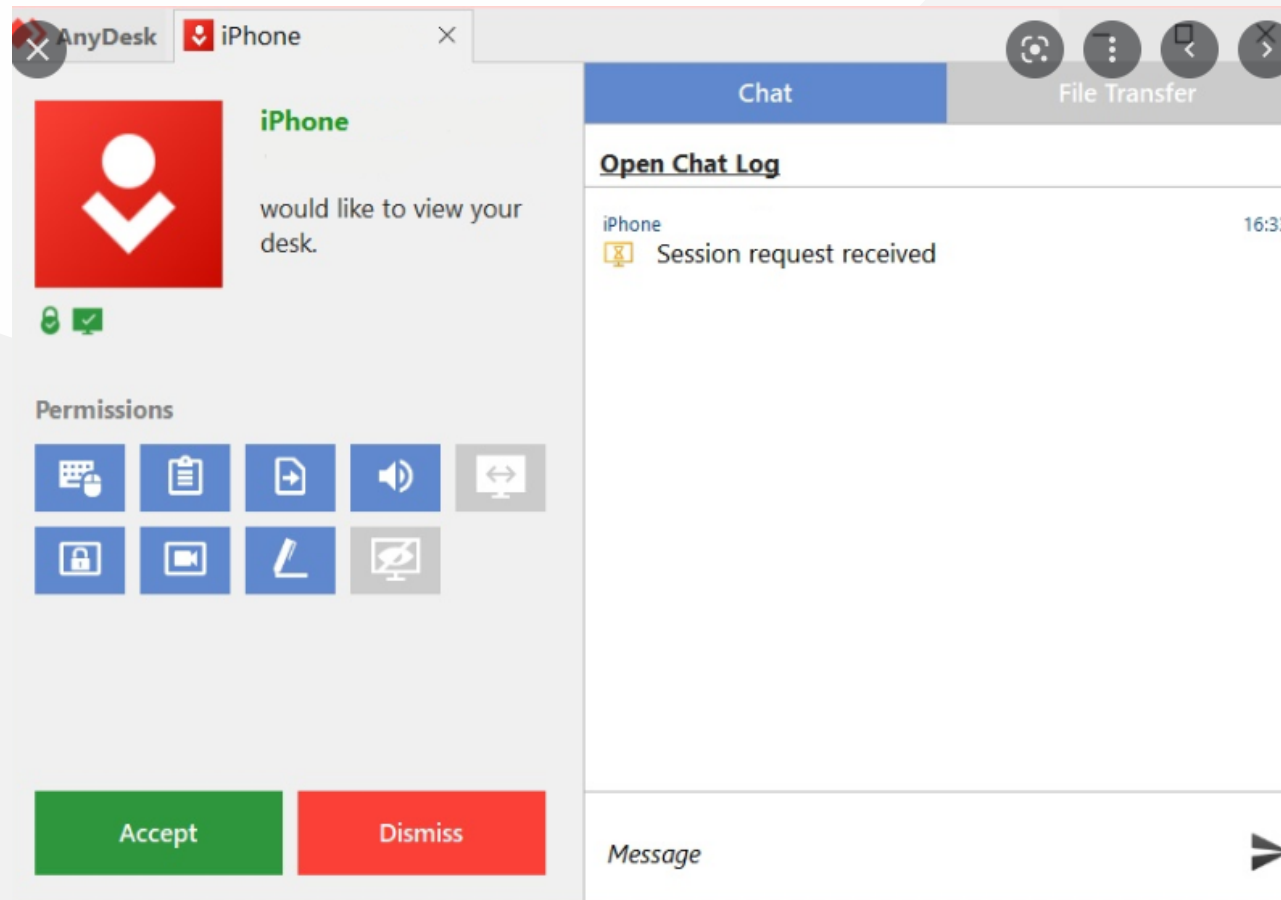
- Remote connection tool
  - TeamViewer – Uzaktan Destek, Uzaktan Erişim, Hizmet Masası, Çevrimiçi İşbirliği ve Toplantılar



The screenshot displays the TeamViewer application window. The title bar shows 'TeamViewer' and standard window controls. The main interface is in Turkish. On the left, a sidebar lists various features: 'Uzaktan Kontrol' (Remote Control), 'Uzaktan Yönetim' (Remote Management), 'Toplantı' (Meeting), 'Bilgisayarlar ve Kişiler' (Computers and People), 'Sohbet' (Chat), 'Arttırılmış Gerçeklik' (Augmented Reality), and 'Daha Fazla Çözüm' (More Solutions) with a 'NEW' badge. The main area is divided into two columns. The left column, titled 'Uzaktan Kontrol İzin Ver' (Allow Remote Control), shows the user's ID as '591 277 070' and the password as 'sbz5bmb3'. The right column, titled 'Uzaktaki Bilgisayarı Kontrol Et' (Control Remote Computer), has a dropdown menu set to 'Uzaktan kontrol' and a text input field for 'Enter Partner ID'. A blue 'Bağlantı' (Connect) button is positioned below the input field. At the bottom, there are two unchecked checkboxes: 'TeamViewer'ı Windows ile birlikte başlat' (Start TeamViewer with Windows) and 'Kolay erişim ver' (Allow easy access). A status bar at the very bottom indicates 'Bağlantı için hazır (güvenli bağlantı)' (Ready for connection (secure connection)).

# AnyDesk

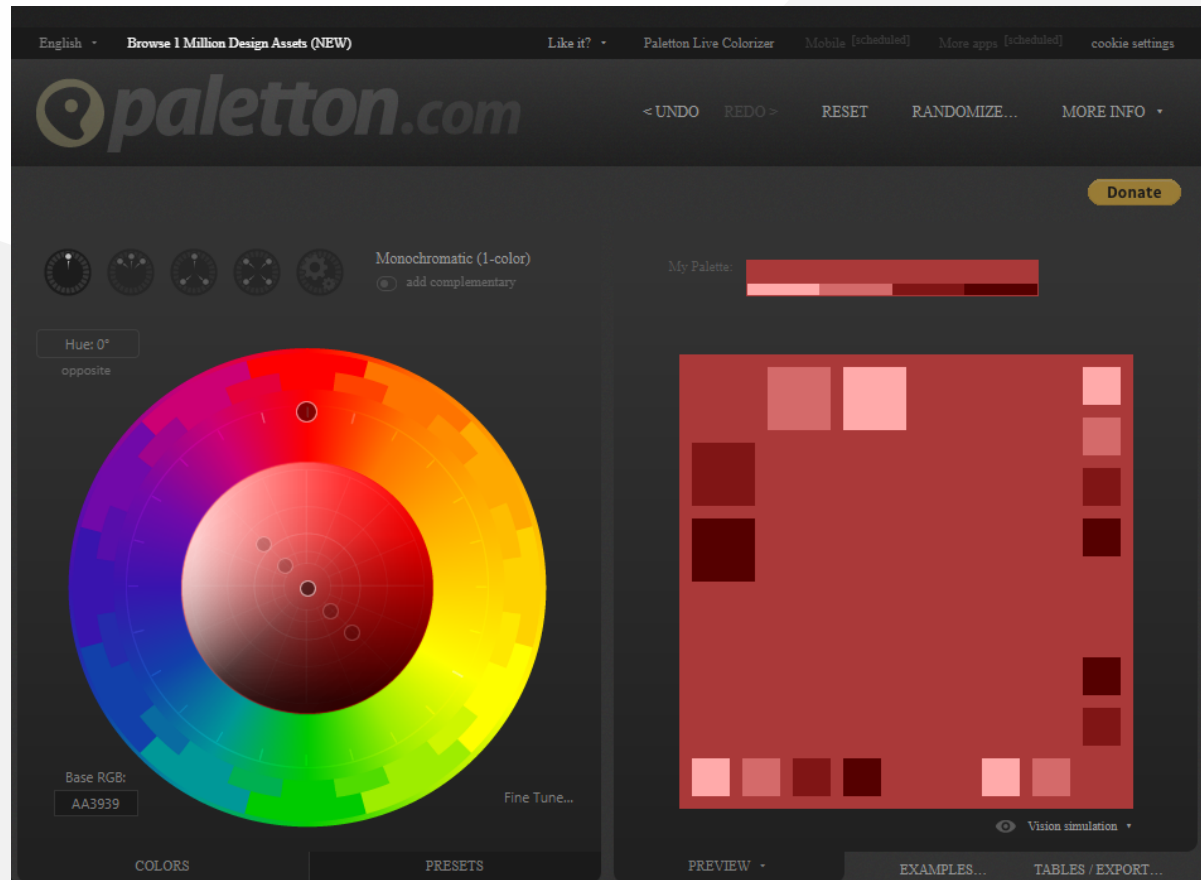
- Remote connection tool
  - The Fast Remote Desktop Application – AnyDesk



# Paletton.com and Colorhunt.co (Color Chooser)

CE103 Algorithms and Programming I

- Generates color palettes and sample usages
  - Paletton - The Color Scheme Designer
  - <https://colorhunt.co/>
  - Also check [Colors Tutorial](#)



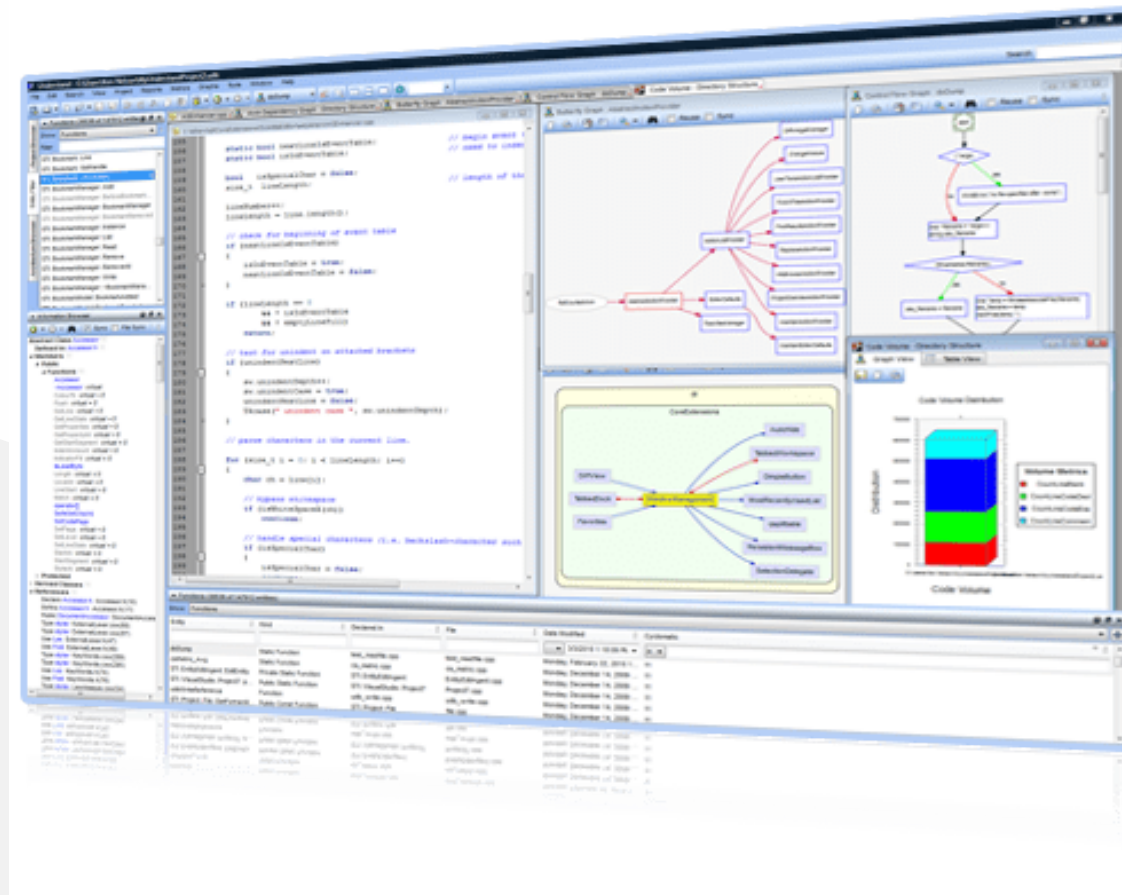
English | Browse 1 Million Design Assets (NEW) | Like it? | Paletton Live Colorizer | cookie settings

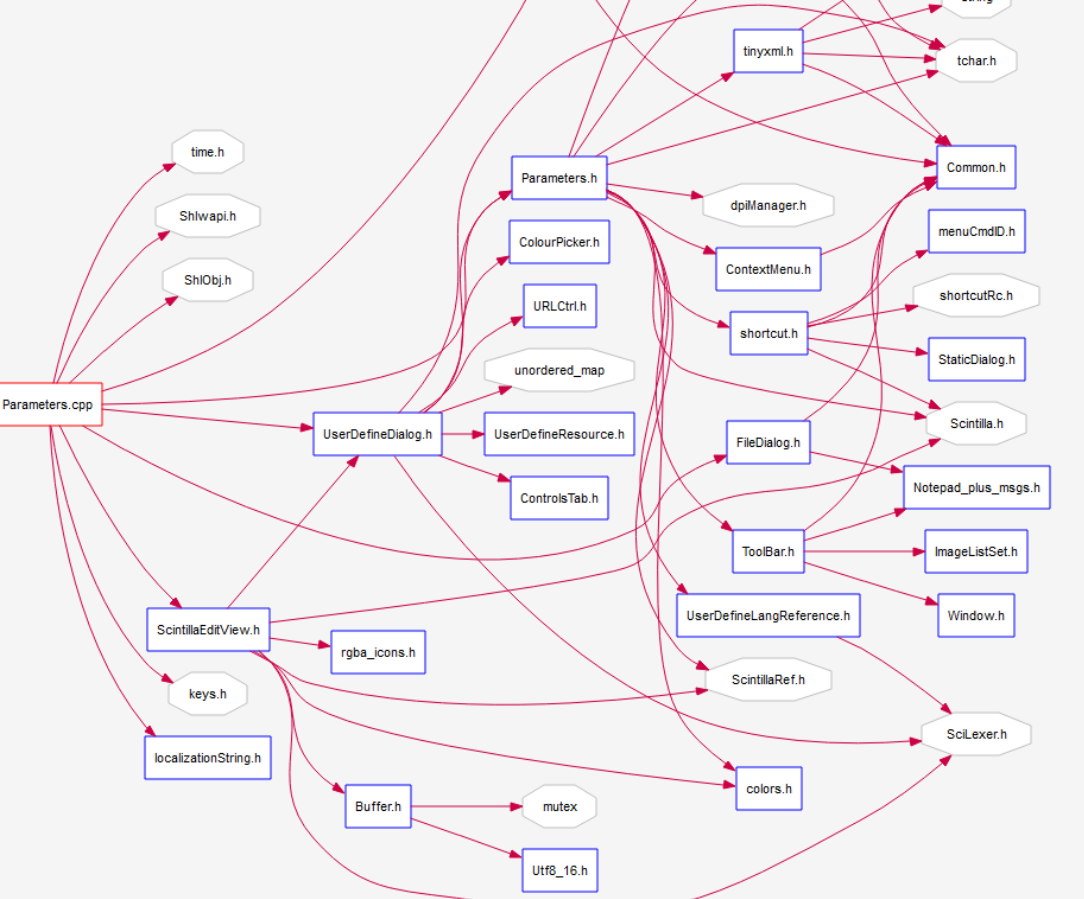
Palette usage examples

The screenshot displays the Paletton Live Colorizer interface. At the top, there are navigation links: "English", "Browse 1 Million Design Assets (NEW)", "Like it?", "Paletton Live Colorizer", and "cookie settings". Below this is a "Palette usage examples" window. The main design area features a dark red header with the text "lorem ipsum" and "DUIS AUTE IRURE DOLOR". Below the header is a horizontal navigation bar with five tabs: "LOREM", "IPSUM", "DOLOR", "SIT AMET", and "ALIQUIP". The main content area is divided into two columns. The left column contains sections for "Mollit Anim", "Lorem", and "Ipsum", each with a brief description and a list of related terms. The right column contains sections for "Lorem ipsum dolor sit amet", "Excepteur sint occaecat", and "Duis aute", each with a description and a "Ut labore..." link. At the bottom of the design area, there is a footer with the text "Copyright © Paletton.com | Lorem | Ipsum | Dolor | Sit amet | Aliquip". On the right side of the interface, there is a sidebar with various options: "Page layout" (White page, Dark page, Positive design, Negative design), "Artwork" (Animated), "Vision simulation", "Fine Tune...", "Randomize...", and a "Close" button.

# Understand (Static Code Analysis)

- <https://emenda.com/scitools-understand/>

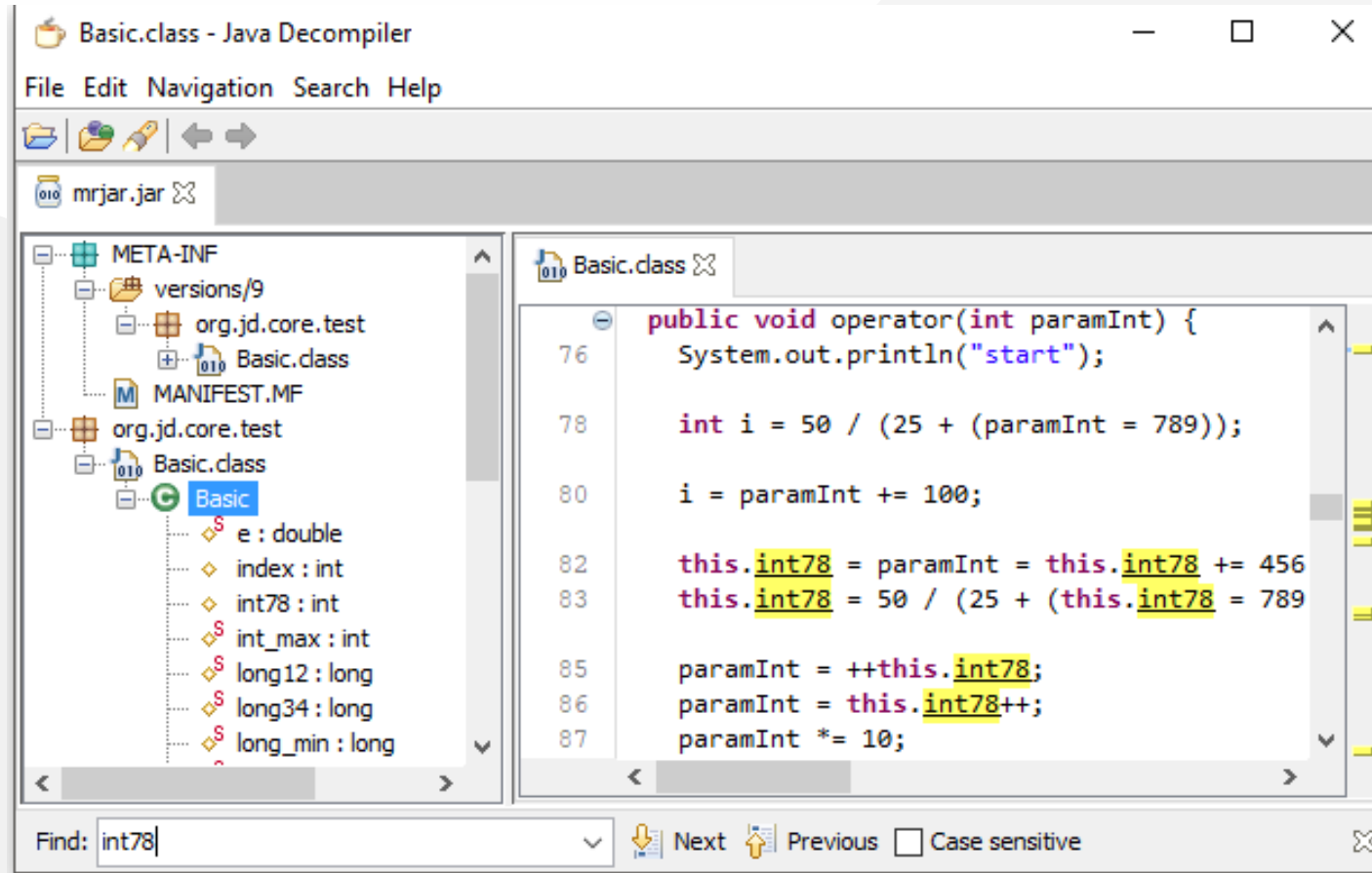




# JD Project (Java Decompiler)

CE103 Algorithms and Programming I

- Java Decompiler for Jar and Class Files, If code is not obfuscated it recover source code from compiled files. Just drag and drop files to GUI
  - <http://java-decompiler.github.io/>
  - You can use it standalone app or with eclipse





# Cutter (Multi-Platform Reverse Engineering Tool)

- Cutter's goal is to be an advanced FREE and open-source reverse-engineering platform while keeping the user experience at mind. Cutter is created by reverse engineers for reverse engineers.
- <https://cutter.re/>

**Cutter**  
Free and Open Source RE Platform powered by Rizin

Download Source

Name	Size
entry.fini0	65
entry.init0	153
entry0	46
fcn.00002000	27
fcn.00002130	41
fcn.00002200	863
fcn.00002630	162
fcn.000026e0	268
fcn.000027f0	4876
fcn.0000344e	3643
fcn.000034e1	4630
fcn.0000355c	4544
fcn.000035ce	4544
fcn.00003b00	6933

Graph (fcn.00004f00)

```
(fcn) fcn.00004f00 128
fcn.00004f00 (int32_t arg1);
; arg int32_t arg1 @ rdi
push r12
push rbp
push rbx
mov rbx, rdi ; arg1
call qword [reloc.fileno] ; [0x8f68:8]=0
mov rdi, rbx
test eax, eax
js 0x4f71

call qword [reloc.__freanding] ; [0x8f88:8]=0
test eax, eax
jne 0x4f50

mov rdi, rbx
call qword [reloc.fileno] ; [0x8f68:8]=0
xor esi, esi
mov edx, 1
mov edi, eax
```

## IDA Pro / Freeware (Native Reverse Engineering Tool)

- IDA Pro as a disassembler is capable of creating maps of their execution to show the binary instructions that are actually executed by the processor in a symbolic representation (assembly language). Advanced techniques have been implemented into IDA Pro so that it can generate assembly language source code from machine-executable code and make this complex code more human-readable.



# IDA Pro / Freeware (Native Reverse Engineering Tool)

The screenshot displays the IDA Pro interface with several key components:

- Functions window:** Lists various subroutines such as sub\_401000, sub\_401240, sub\_401590, sub\_4018C0, and sub\_401900.
- Assembly view:** Shows assembly instructions for several locations:
  - loc\_401726: `mov ecx, [ebp+arg_0]`, `mov eax, [ecx]`, `mov ecx, [eax+4]`, `mov di, [eax+ecx+40h]`, `mov ecx, [eax+ecx+30h]`, `mov [ebp+var_11], di`, `mov eax, [ecx+20h]`, `cmp dword ptr [eax], 0`, `jz short loc_401764`
  - loc\_401760: `mov [ebp+var_4], 1`
  - loc\_401767: `mov ebx, [ebp+var_20]`
  - loc\_401761: `mov edx, [ecx+30h]`, `mov eax, [edx]`, `test eax, eax`, `jle short loc_401761`
  - loc\_40176A: `mov edi, [ebp+arg_0]`, `mov eax, edi`, `mov eax, [eax+4]`, `mov dword ptr [eax+edi+20h], 0`, `mov dword ptr [eax+edi+30h], 0`, `mov eax, [edi]`, `mov ecx, [eax+4]`, `add ecx, edi`, `test esi, esi`, `jz loc_401800`
- Graph overview:** A small thumbnail of the control flow graph.
- Strings window:** Lists strings found in the binary, such as "Could not encrypt", "Level 1 cleared, Good Rabbit d", "Nope!", "invalid string position", "string too long", "bad cast", "bad allocation", "permission denied", "file exists", "no such device", "filename too long", "device or resource busy", "io error", "directory not empty", "invalid argument", "no such file or device", "no such file or directory", "function not supported", "no lock available".
- Imports window:** Lists imported functions from the kernel32.dll library, including GetProcAddress, LoadResource, LockResource, SetConsoleTextAttribute, GetLastError, ReadConsole, WriteConsole, SetStdHandle, LoadLibraryW, FlushConsoleInputBuffer, GetTickCount, CreateProcessW, Sleep, CloseHandle, GetCurrentPathW, WriteFile, CreateFileW, and FindResourceA.

# Code Visualization (Python, C , C++ , Java)

- This coding tutor tool helps you learn Python, JavaScript, C, C++, and Java by visualizing code execution.
  - <https://pythontutor.com/>

Python 3.6

```

1 def listSum(numbers):
2   if not numbers:
3     return 0
4   else:
5     (f, rest) = numbers
6     return f + listSum(rest)
7
8 myList = (1, (2, (3, None)))
9 total = listSum(myList)

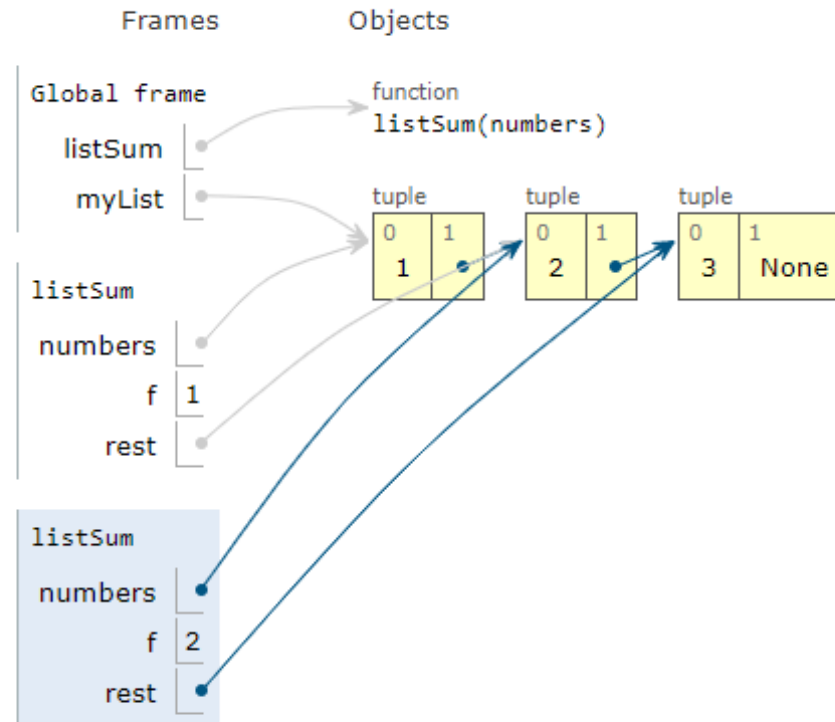
```

[Edit this code](#)

→ line that just executed  
 → next line to execute

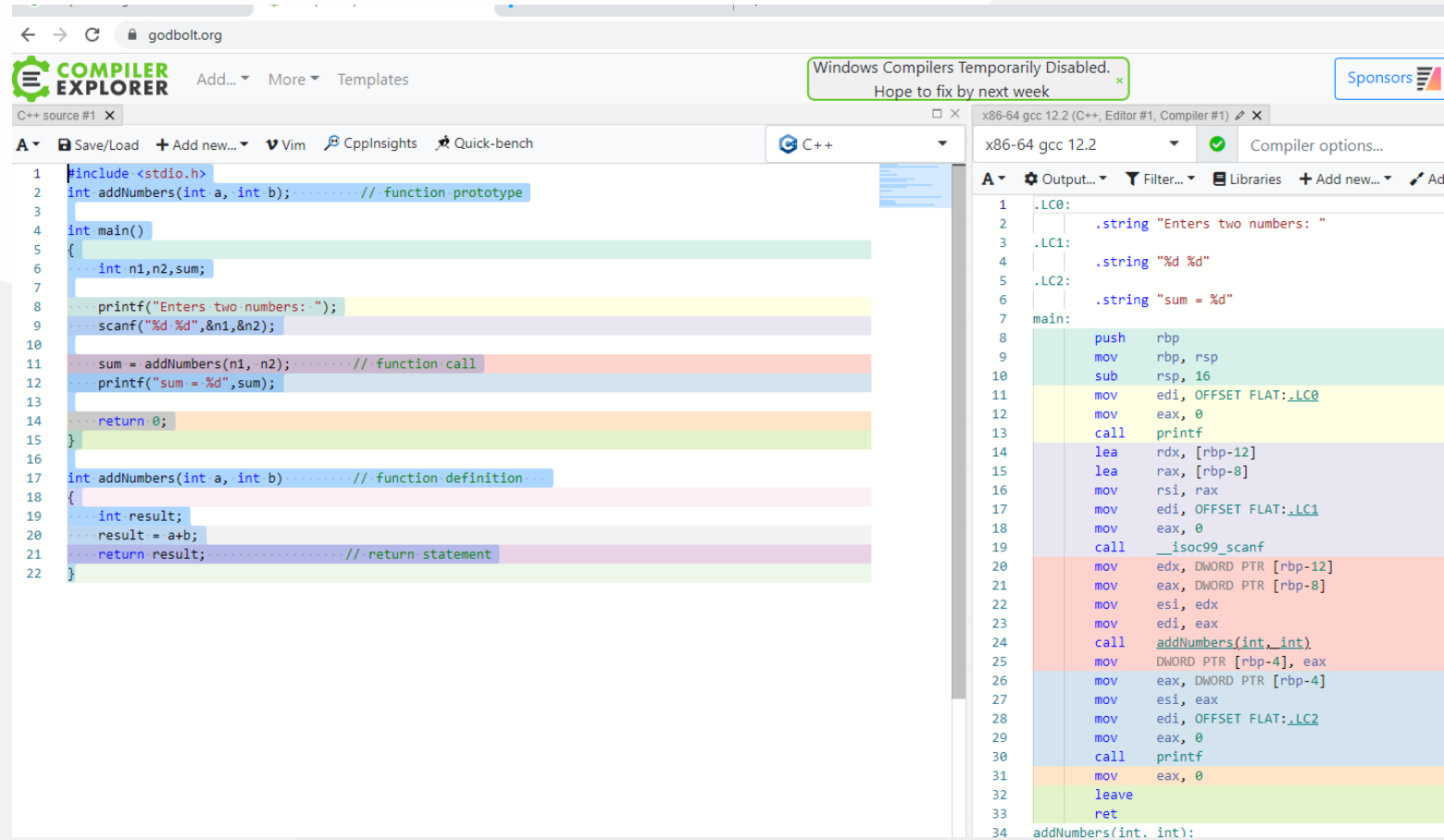
Step 11 of 22

Visualized using [Python Tutor](#)  
[Customize visualization](#)



# Assembly of C Code

- Multilanguage supported. Convert source code to assembly codes
  - <https://godbolt.org/>



The screenshot displays the Godbolt Compiler Explorer interface. On the left, the C++ source code is shown, including a function prototype for `addNumbers`, a `main` function that takes two integers and prints their sum, and the implementation of `addNumbers`. On the right, the corresponding assembly code for x86-64 GCC 12.2 is displayed. The assembly shows the stack frame setup, the `printf` call for the first prompt, the `scanf` call to read two integers, the `call` to `addNumbers`, and the final `printf` call to output the sum. A notification at the top indicates that Windows compilers are temporarily disabled.

```
1 #include <stdio.h>
2 int addNumbers(int a, int b); // function prototype
3
4 int main()
5 {
6     int n1,n2,sum;
7
8     printf("Enters two numbers: ");
9     scanf("%d %d",&n1,&n2);
10
11     sum = addNumbers(n1, n2); // function call
12     printf("sum = %d",sum);
13
14     return 0;
15 }
16
17 int addNumbers(int a, int b) // function definition...
18 {
19     int result;
20     result = a+b;
21     return result; // return statement
22 }
```

```
1 .LC0:
2     .string "Enters two numbers: "
3 .LC1:
4     .string "%d %d"
5 .LC2:
6     .string "sum = %d"
7 main:
8     push    rbp
9     mov     rbp, rsp
10    sub     rsp, 16
11    mov     edi, OFFSET FLAT:.LC0
12    mov     eax, 0
13    call   printf
14    lea    rdx, [rbp-12]
15    lea    rax, [rbp-8]
16    mov     rsi, rax
17    mov     edi, OFFSET FLAT:.LC1
18    mov     eax, 0
19    call   __isoc99_scanf
20    mov     edx, DWORD PTR [rbp-12]
21    mov     eax, DWORD PTR [rbp-8]
22    mov     esi, edx
23    mov     edi, eax
24    call   addNumbers(int, int)
25    mov     DWORD PTR [rbp-4], eax
26    mov     eax, DWORD PTR [rbp-4]
27    mov     esi, eax
28    mov     edi, OFFSET FLAT:.LC2
29    mov     eax, 0
30    call   printf
31    mov     eax, 0
32    leave
33    ret
34 addNumbers(int, int):
```

## Mobile Device Screen Sharing for Demo

- Show USB or Wifi connected Mobile Device Screen on PC and Provide Controls
  - [GitHub - Genymobile/scrcpy: Display and control your Android device](#)
  - Run `scrcpy` for single mobile phone.
  - [Open Source Project - Scrcpy now works wirelessly](#)

## Travis-CI

- Travis-CI is a continues integration platform
- Travis-CI free option removed for this reason, its not in our scope.
- It uses Travis.yml files for actions.

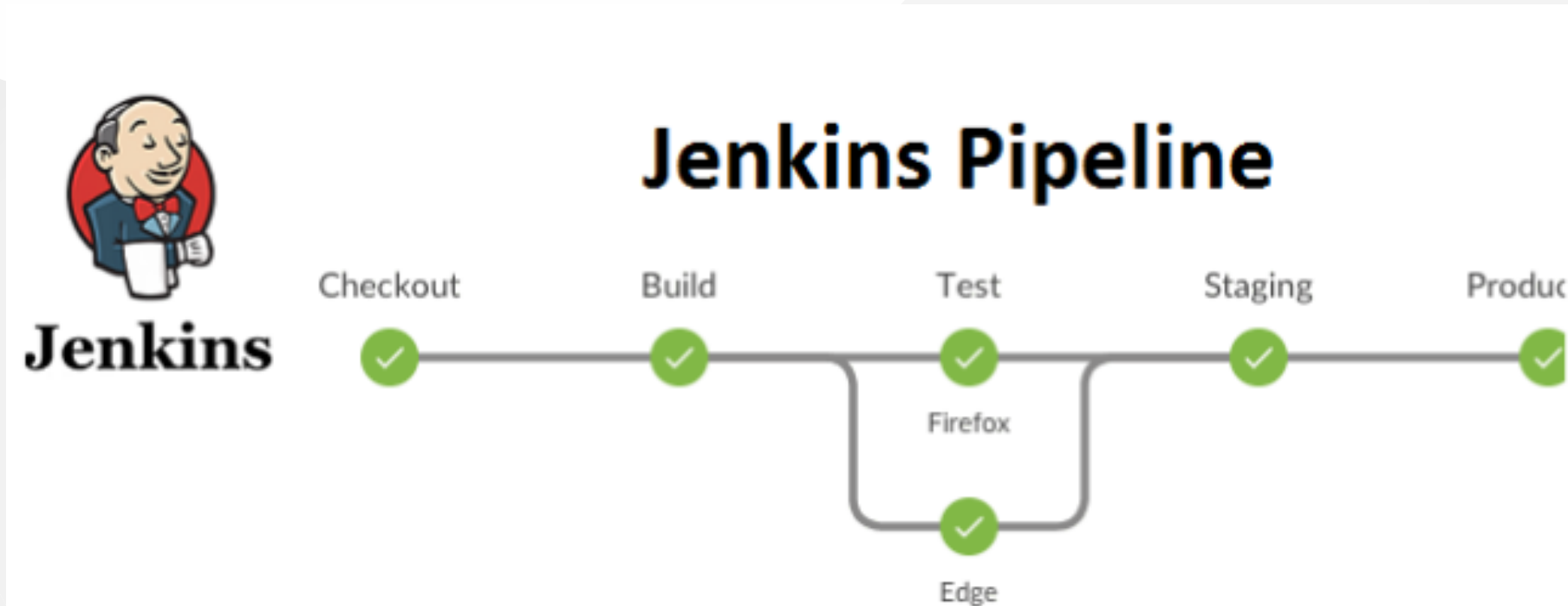


## AppVeyor

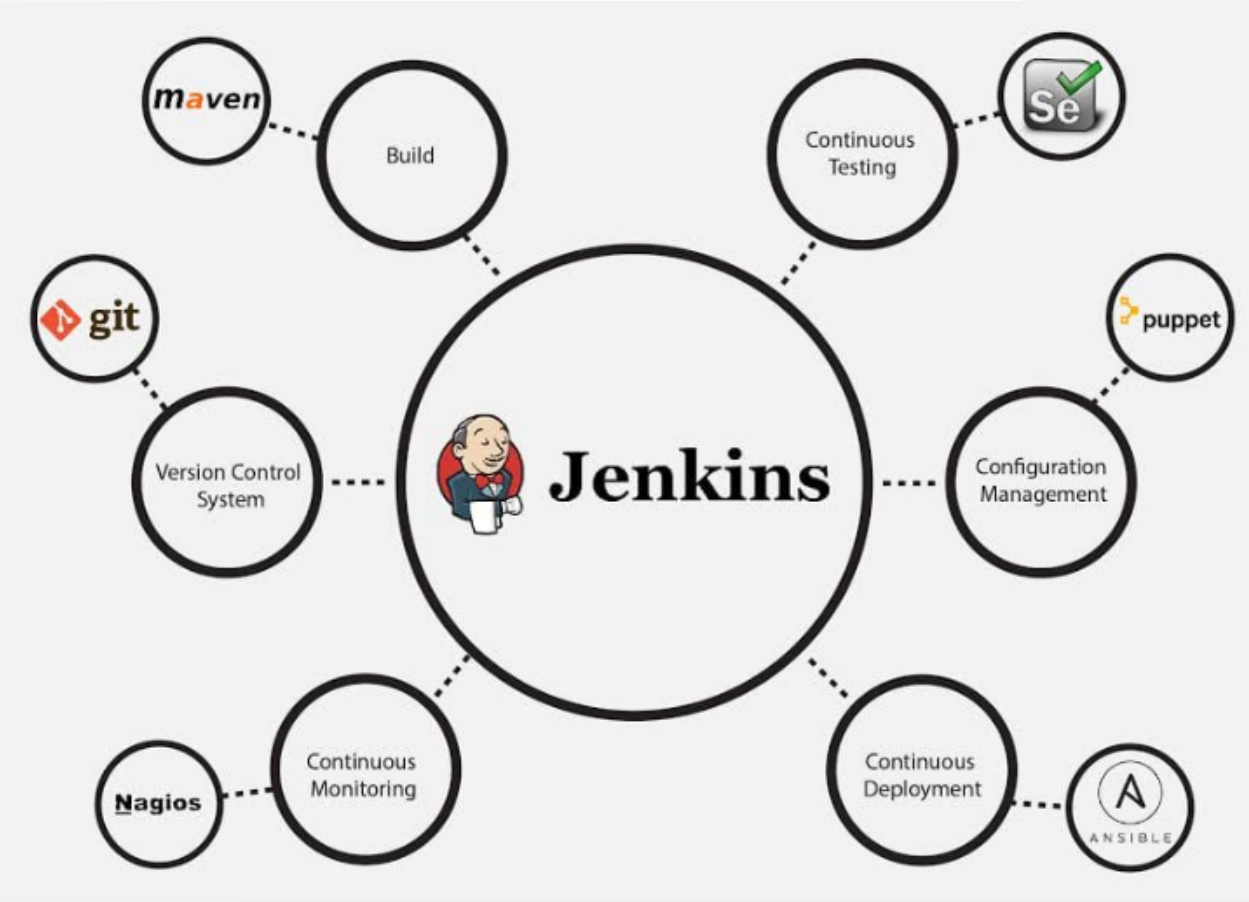
- Another CI platform it has free option for public builds.
  - <https://www.appveyor.com>
  - [GitHub - Kimserey/hello-world-nuget](#)
  - [hello-world-nuget/appveyor.yml at master · Kimserey/hello-world-nuget · GitHub](#)

## Jenkins

- Self-hosted solution for CI operations, Has integration with Github and several platforms.
  - <https://www.jenkins.io/>
  - <https://www.jenkins.io/doc/pipeline/tour/hello-world/>



# Jenkins



## Jenkins

- <https://www.jenkins.io/solutions/github/>

### Configure Global Security

Enable security

TCP port for JNLP slave agents  Fixed :   Random  Disable

Disable remember me

Access Control

#### Security Realm

- Delegate to servlet container
- Github Authentication Plugin

#### Global Github OAuth Settings

GitHub Web URI

GitHub API URI

Client ID

Client Secret

OAuth Scope(s)

- Jenkins' own user database
- LDAP
- Unix user/group database

#### Authorization

- Anyone can do anything
- Github Committer Authorization Strategy

#### Github Authorization Settings

Admin User Names

Participant in Organization

Use Github repository permissions

Grant READ permissions to all Authenticated Users

Grant CREATE Job permissions to all Authenticated Users

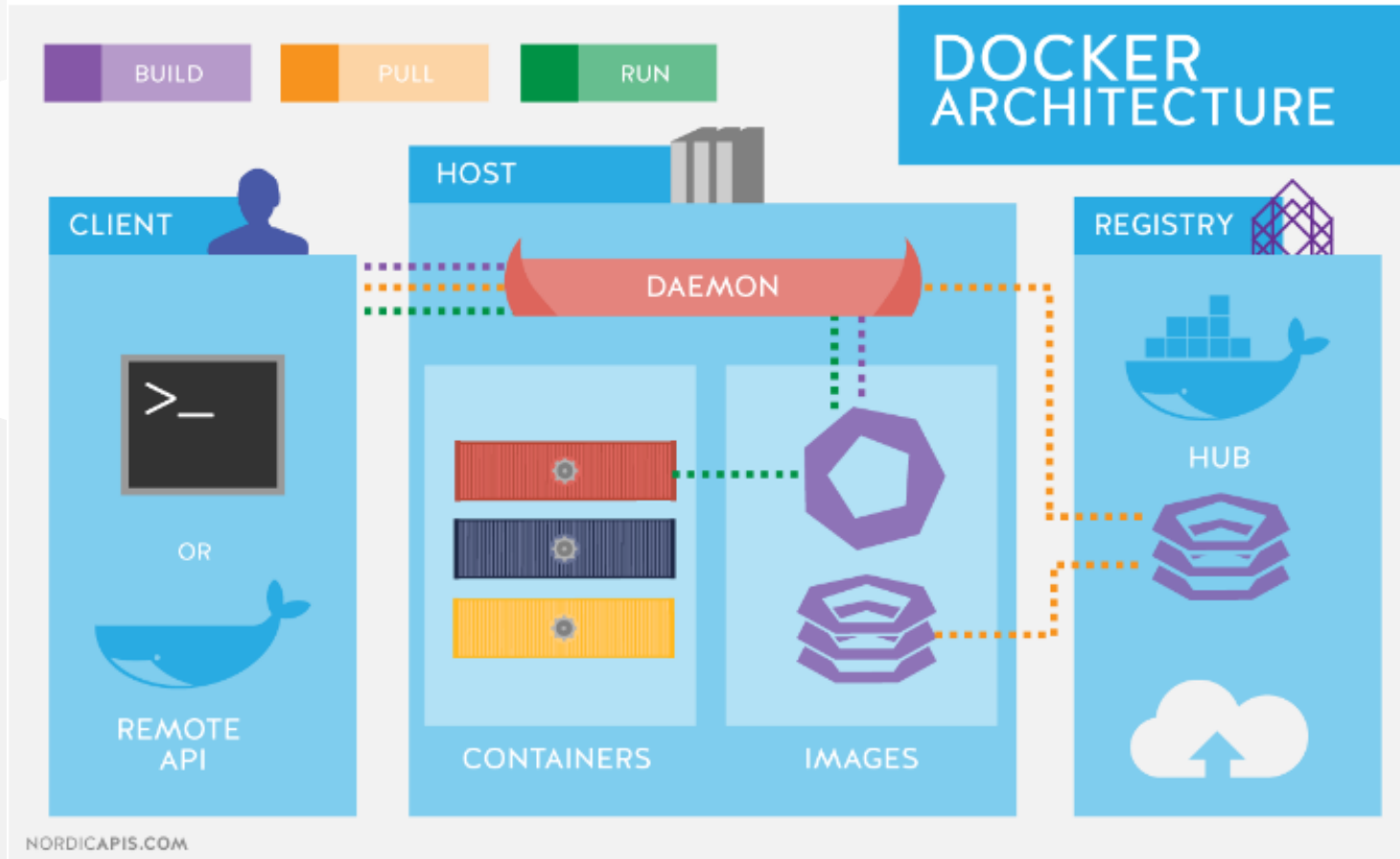
## Vagrant

- Vagrant is a tool for building and managing virtual machine environments in a single workflow. With an easy-to-use workflow and focus on automation, Vagrant lowers development environment setup time, increases production parity, and makes the "works on my machine" excuse a relic of the past.
  - <https://www.vagrantup.com/>
- Setup Development Environment with Vagrant
  - [Setting Up Development Environment Using Vagrant - Edureka](#)
  - [development-environment/Vagrantfile at master · gantsign/development-environment · GitHub](#)

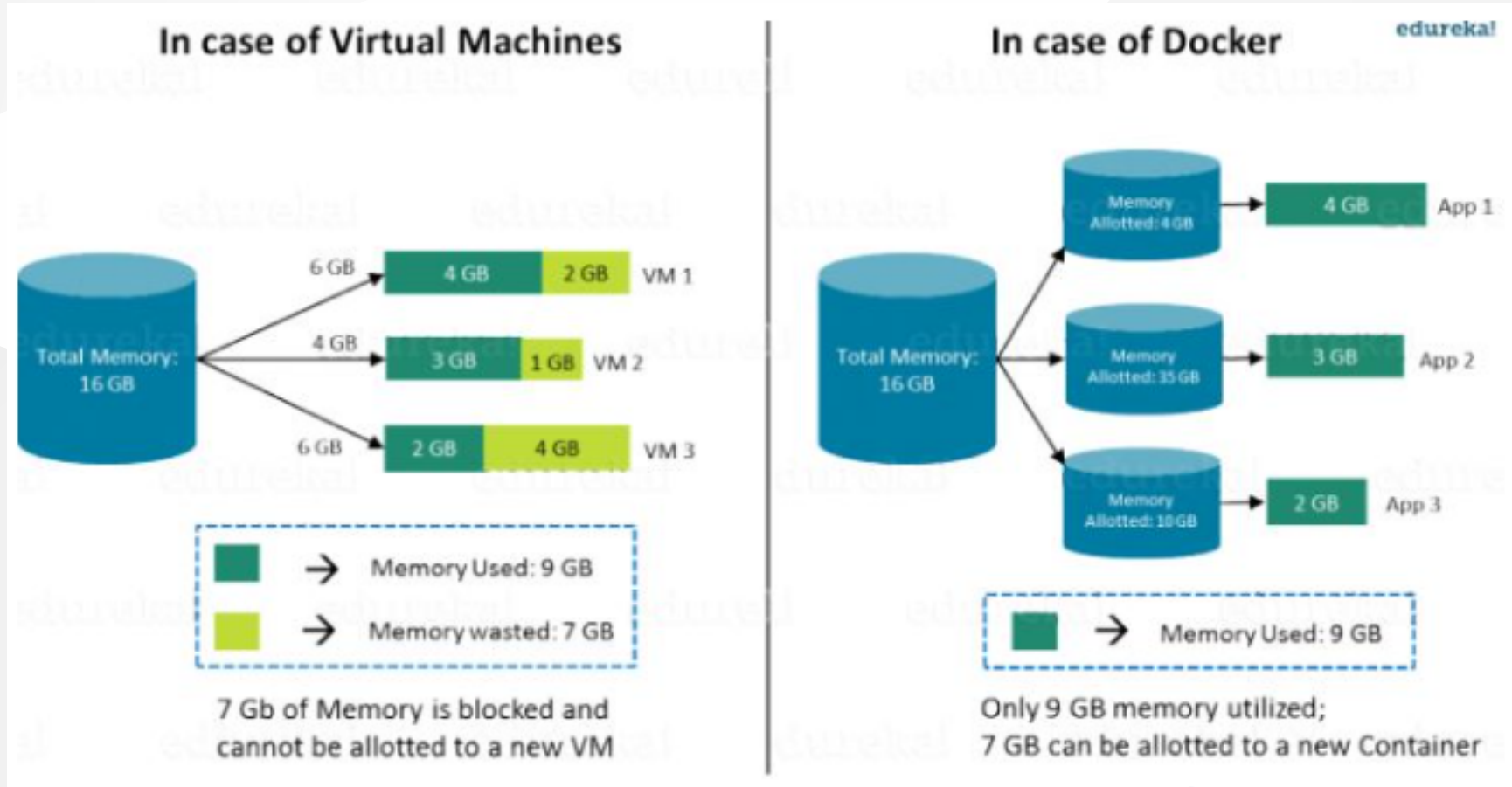
## Docker / Docker Compose / Kubernetes (1)

- Docker takes away repetitive, mundane configuration tasks and is used throughout the development lifecycle for fast, easy and portable application development – desktop and cloud.
  - [https://www.youtube.com/watch?v=nBwJm0onzeo&ab\\_channel=GaryExplains](https://www.youtube.com/watch?v=nBwJm0onzeo&ab_channel=GaryExplains) Dockerfile
  - <https://devopedia.org/docker>
- DockerHub
- Docker Compose Yaml
- Dockerrun.aws.json (AWS)

## Docker / Docker Compose / Kubernetes (2)

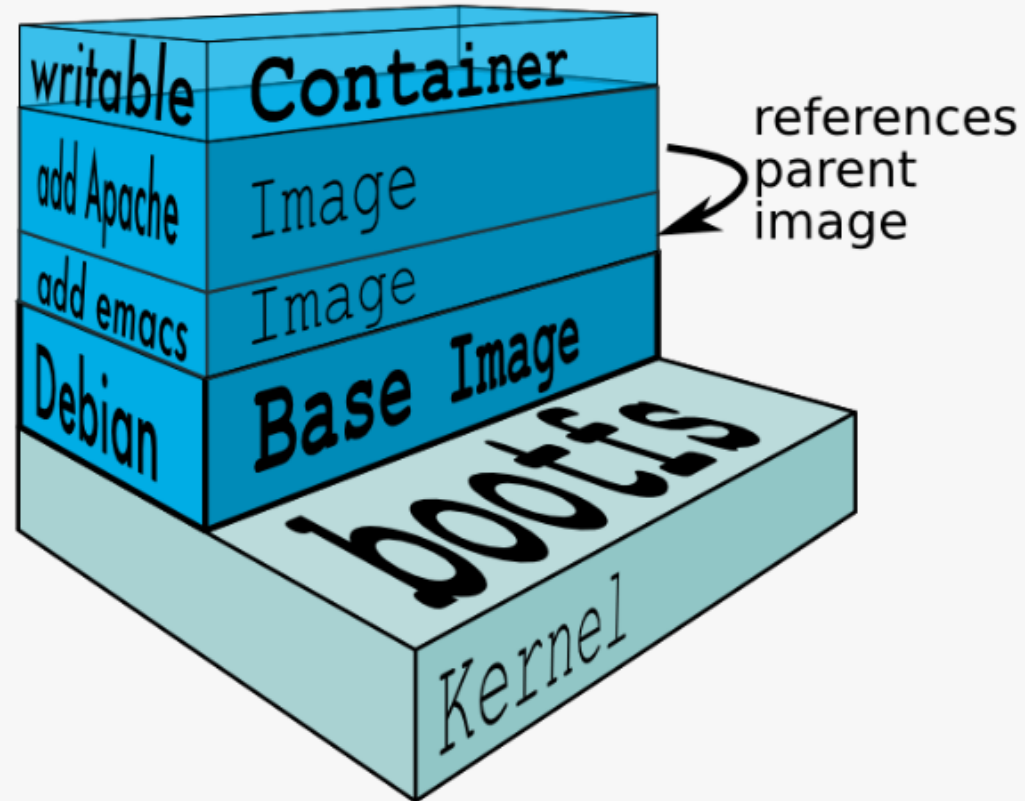


# Docker / Docker Compose / Kubernetes (3)





## Docker / Docker Compose / Kubernetes (4)



## Docker / Docker Compose / Kubernetes (5)

```
FROM node:9.3.0-alpine

RUN npm install -g @angular/cli@1.5.5 \
    && mkdir -p /usr/src/pintail-whoami

WORKDIR /usr/src/pintail-whoami

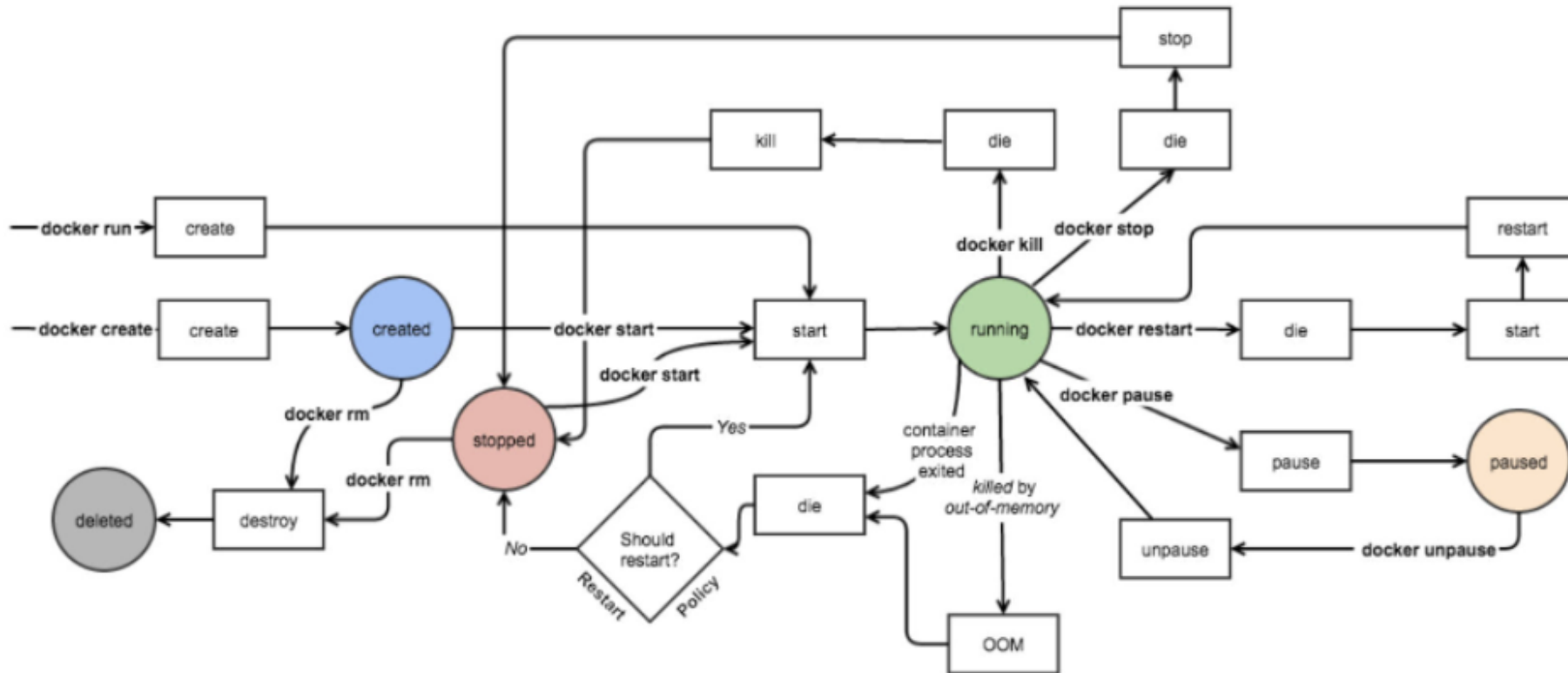
ADD . /usr/src/pintail-whoami

RUN npm install && ng build

EXPOSE 80

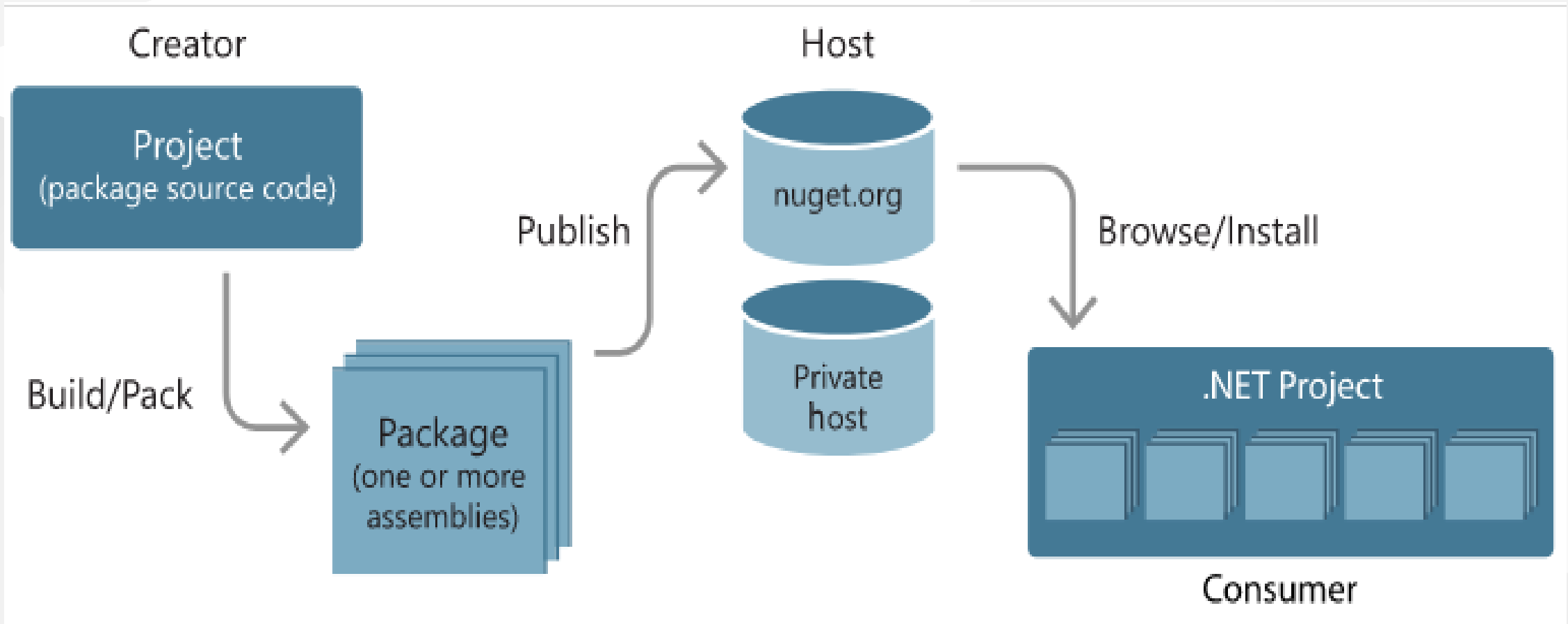
CMD node server.js $HOSTNAME
```

## Docker / Docker Compose / Kubernetes (6)



## NuGet Packages (1)

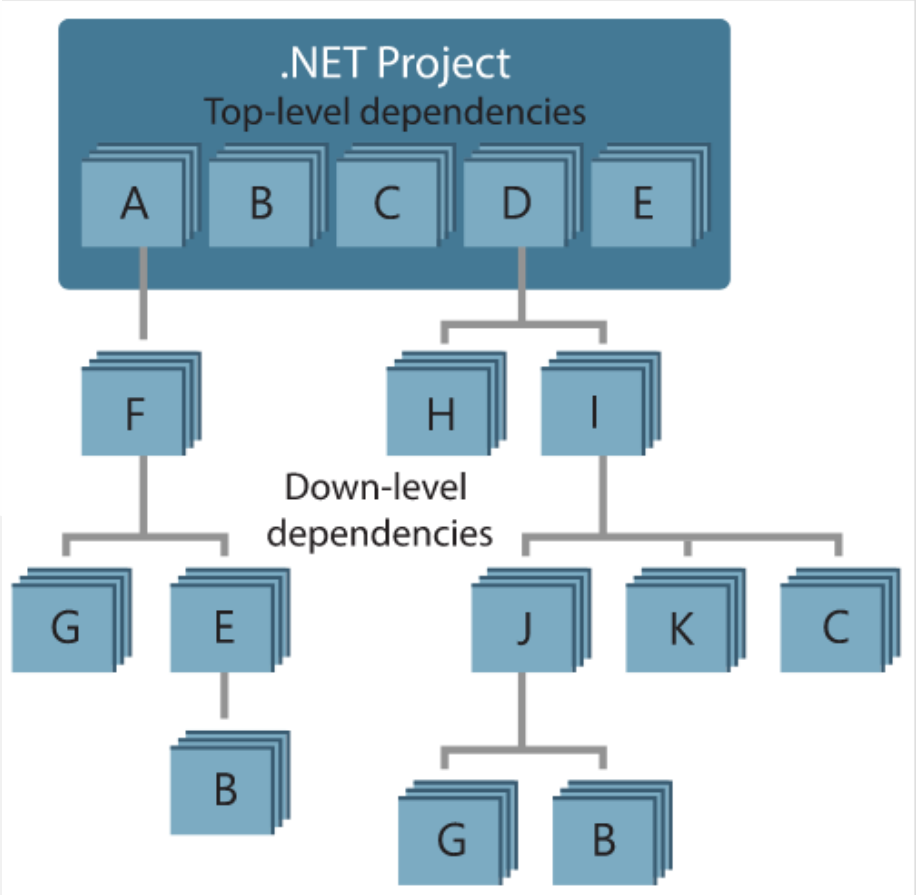
- <https://www.nuget.org/packages>
- [What is NuGet and what does it do? | Microsoft Learn](#)



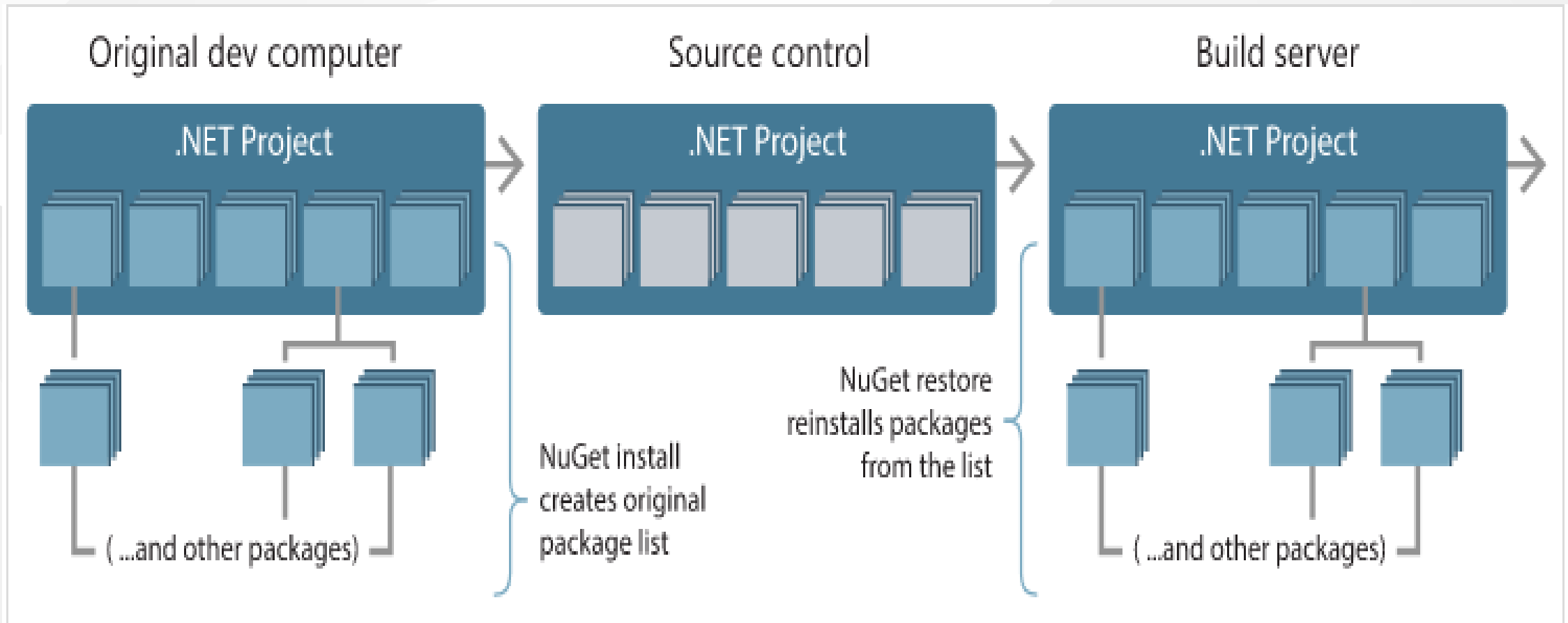
## NuGet Tools (2)

Tool	Platforms	Applicable Scenarios	Description
<a href="#">dotnet CLI</a>	All	Creation, Consumption	CLI tool for .NET Core and .NET Standard libraries, and for SDK-style projects that target .NET Framework (see <a href="#">SDK attribute</a> ). Provides certain NuGet CLI capabilities directly within the .NET Core tool chain. As with the <code>nuget.exe</code> CLI, the dotnet CLI does not interact with Visual Studio projects.
<a href="#">nuget.exe CLI</a>	All	Creation, Consumption	CLI tool for .NET Framework libraries and non-SDK-style projects that target .NET Standard libraries. Provides all NuGet capabilities, with some commands applying specifically to package creators, some applying only to consumers, and others applying to both. For example, package creators use the <code>nuget pack</code> command to create a package from various assemblies and related files, package consumers use <code>nuget install</code> to include packages in a project folder, and everyone uses <code>nuget config</code> to set NuGet configuration variables. As a platform-agnostic tool, the NuGet CLI does not interact with Visual Studio projects.
<a href="#">Package Manager Console</a>	Visual Studio on Windows	Consumption	Provides <a href="#">PowerShell commands</a> for installing and managing packages in Visual Studio projects.
<a href="#">Package Manager UI</a>	Visual Studio on Windows	Consumption	Provides an easy-to-use UI for installing and managing packages in Visual Studio projects.
<a href="#">Manage NuGet UI</a>	Visual Studio for Mac	Consumption	Provide an easy-to-use UI for installing and managing packages in Visual Studio for Mac projects.
<a href="#">MSBuild</a>	Windows	Creation, Consumption	Provides the ability to create packages and restore packages used in a project directly through the MSBuild tool chain.

# Managing dependencies (3)

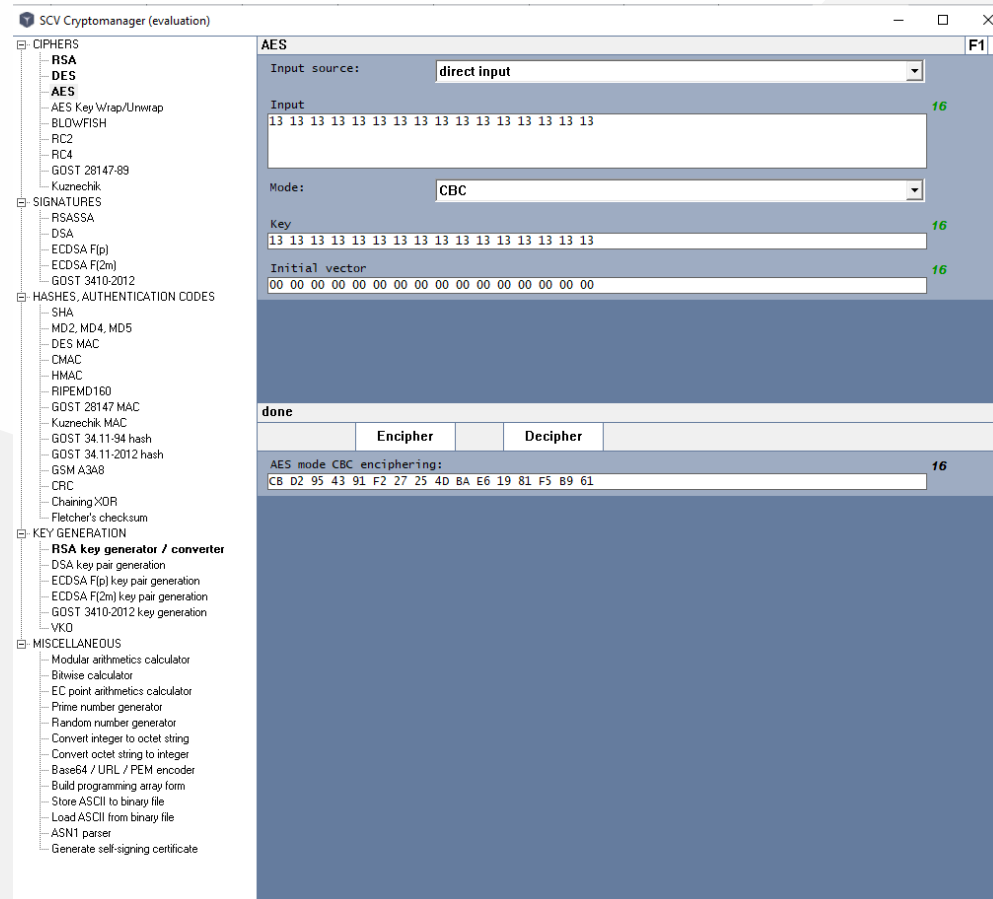


## Tracking references and restoring packages (4)



# SCV Cryptomanager

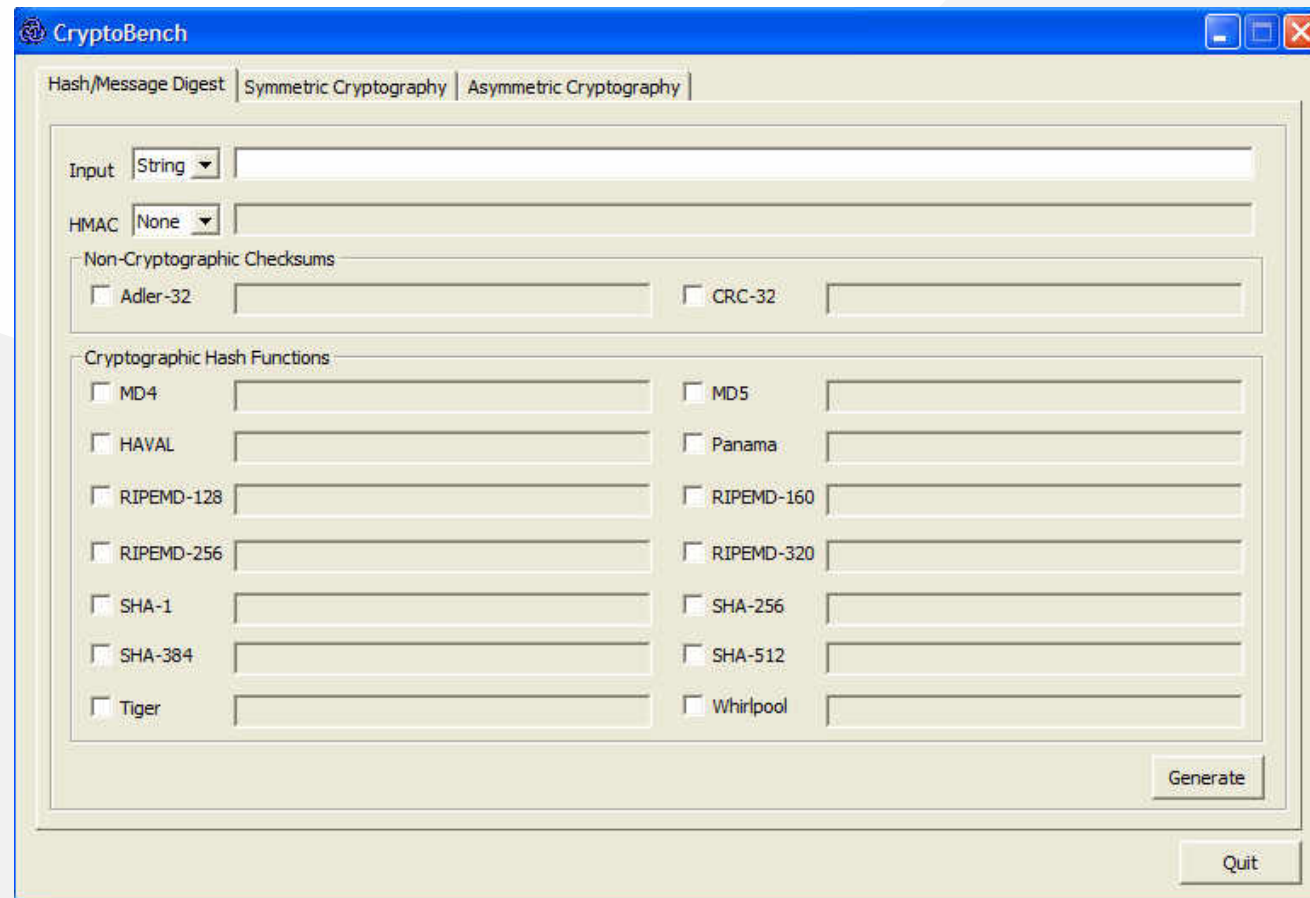
- SCV Crypto Manager has several tools for cryptographic operations.
  - <https://cryptomanager.com/download.php>





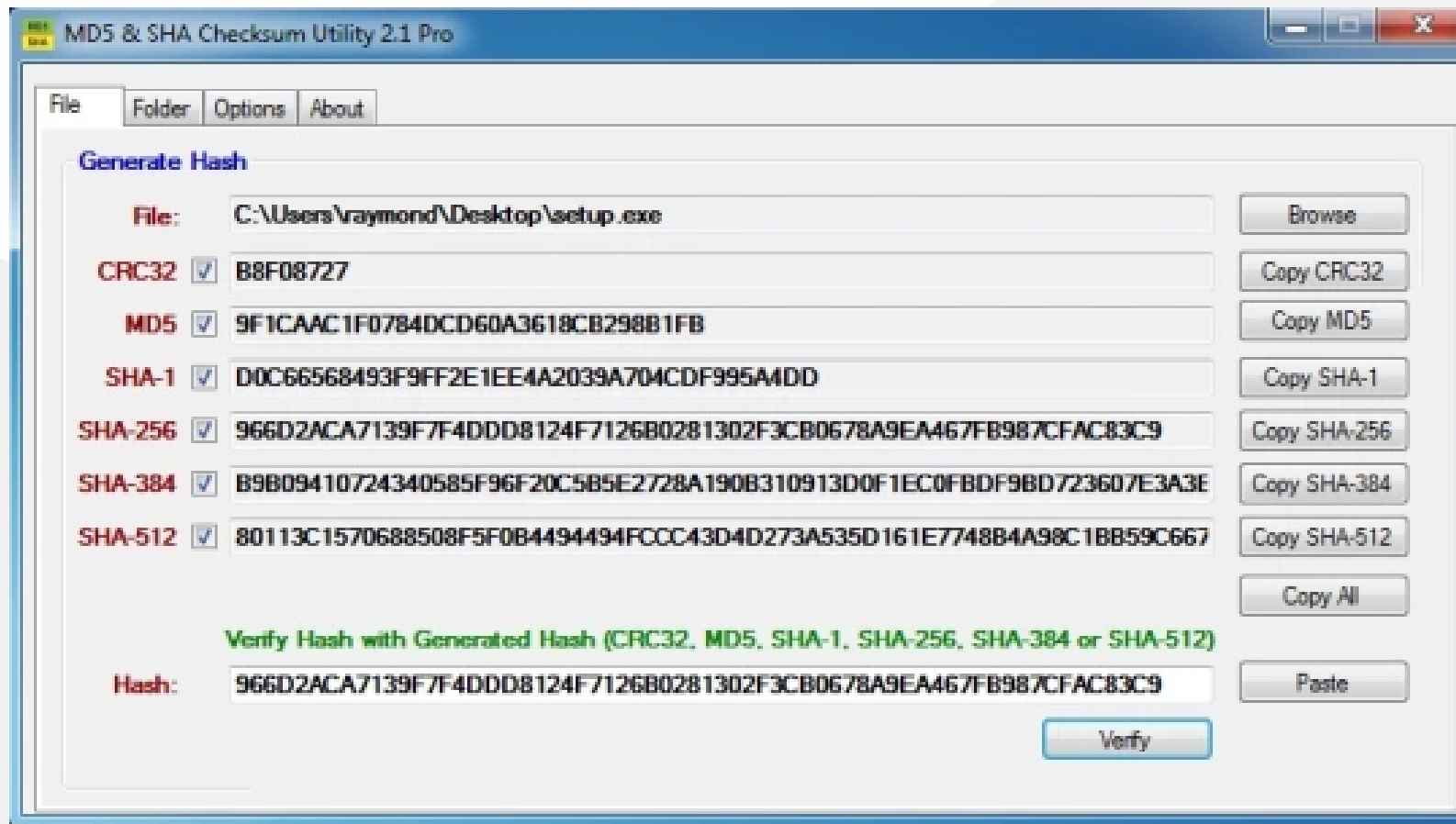
## Addario CryptoBench

- CryptoBench can be used for hash and symmetric asymmetric encryption-decryption operations.
  - [CryptoBench Download Page](#)
  - <http://www.addario.org/files/CryptoBench v1.0.1.zip>



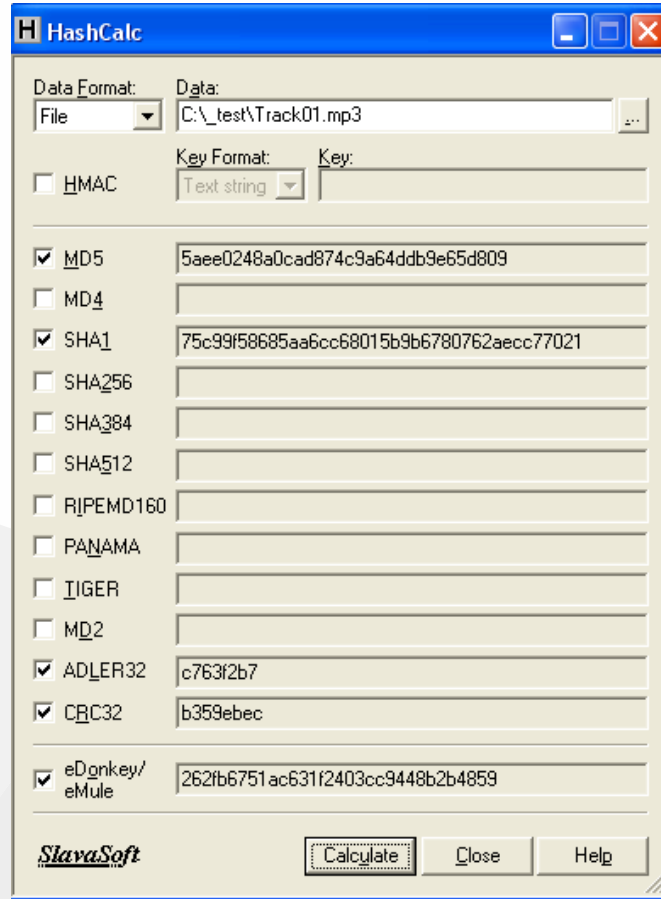
# Raymond's MD5 & SHA Checksum Utility

- Hash Calculation Utility
- [MD5 & SHA Checksum Utility | Raymond's WordPress](#)



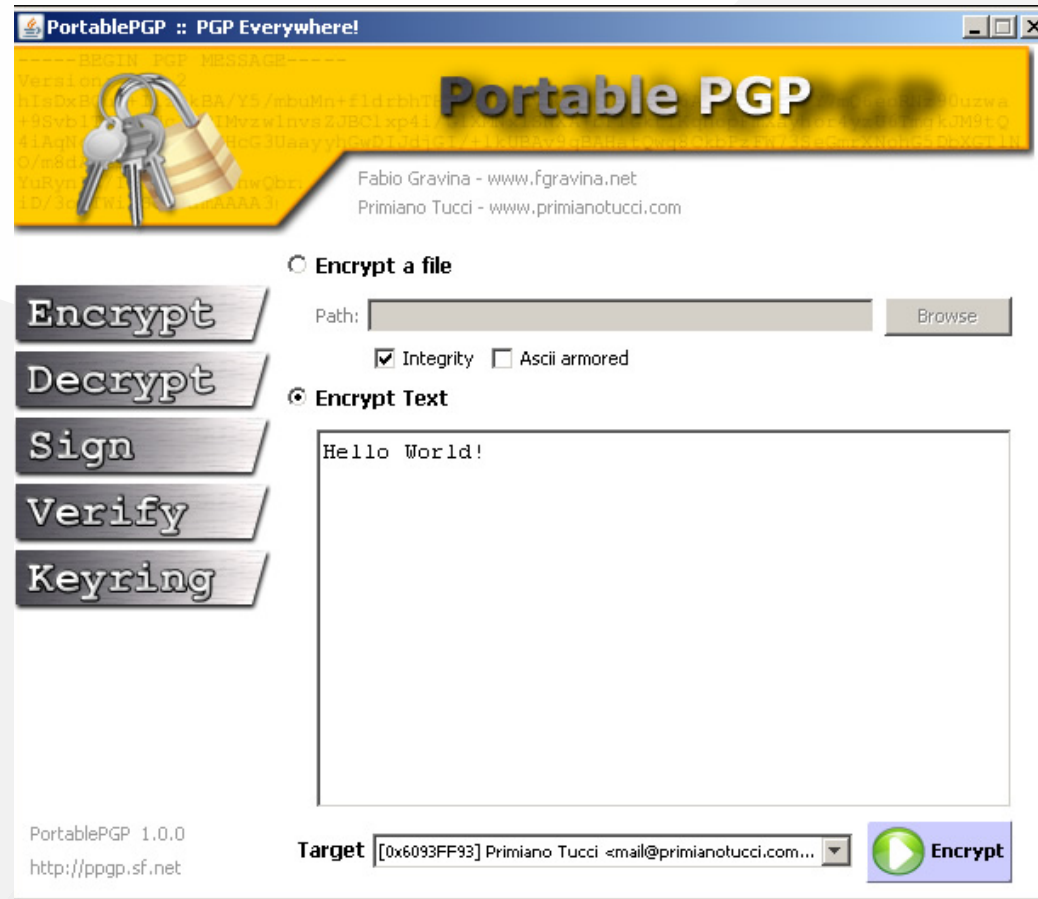
## SlavaSoft HashCalc

- SlavaSoft HashCalc - Hash, CRC, and HMAC Calculator



# Portable PGP

- Portable PGP uses for the generation of PGP keys to transfer files securely via e-mail or other channels. You can encrypt or sign your documents with this tool then the receiver can open or verify your e-mail.
- <https://ppgp.sourceforge.net/>



## Online Programming Environments

- Hackerrank
  - <https://www.hackerrank.com/>
- CS50 Sandbox
  - <https://sandbox.cs50.io/>
- Programiz C Online Compiler
  - [Online C Compiler](#)

*End – Of – Week – 2*