

C Developer

Foundation and First Steps



Course Objectives

- ✓ Install tools
- ✓ Compile your first program



Course Plan

1. C Language and Programs
2. Tools
3. Program Creation

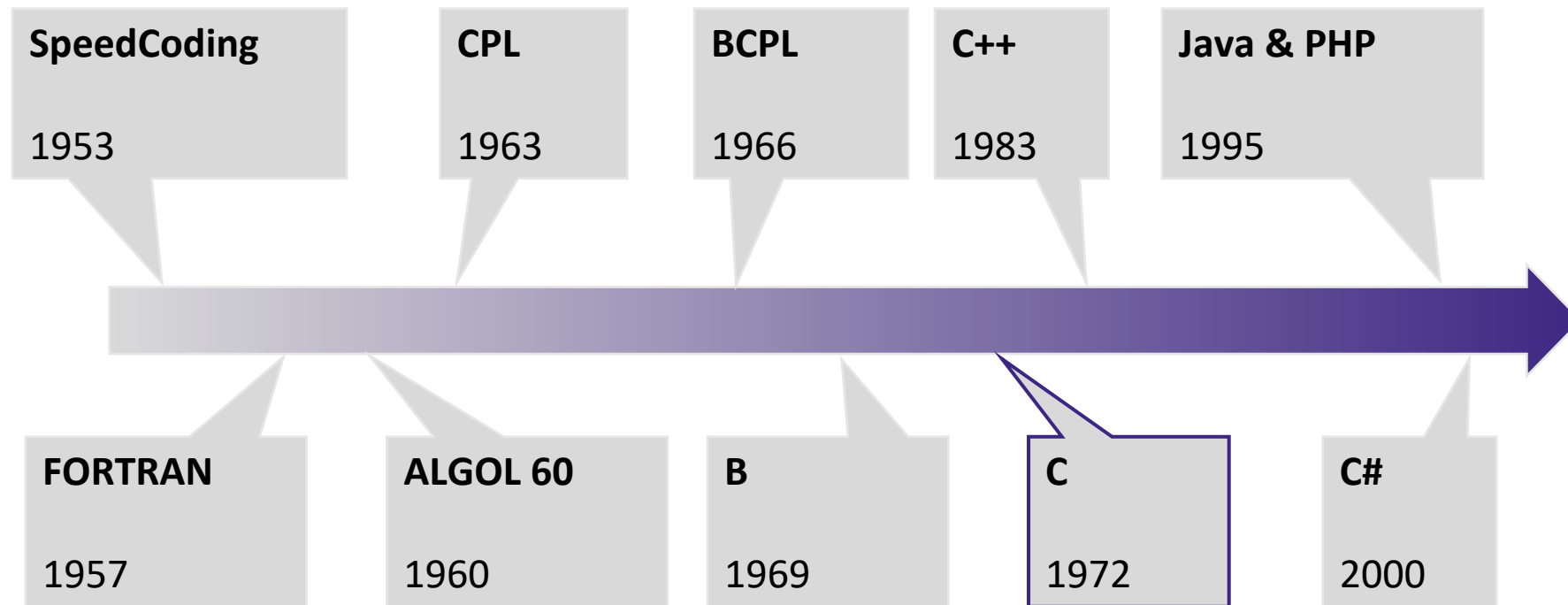


1. C Language and Programs



1. C Language and Programs

History



1. C Language and Programs

History

1969 – Ken Thompson, author of the B Language is joined by Dennis Ritchie

1972 – Dennis Ritchie added arrays, pointers, structures, *etc.*

1973 – UNIX is rewritten in C



1. C Language and Programs

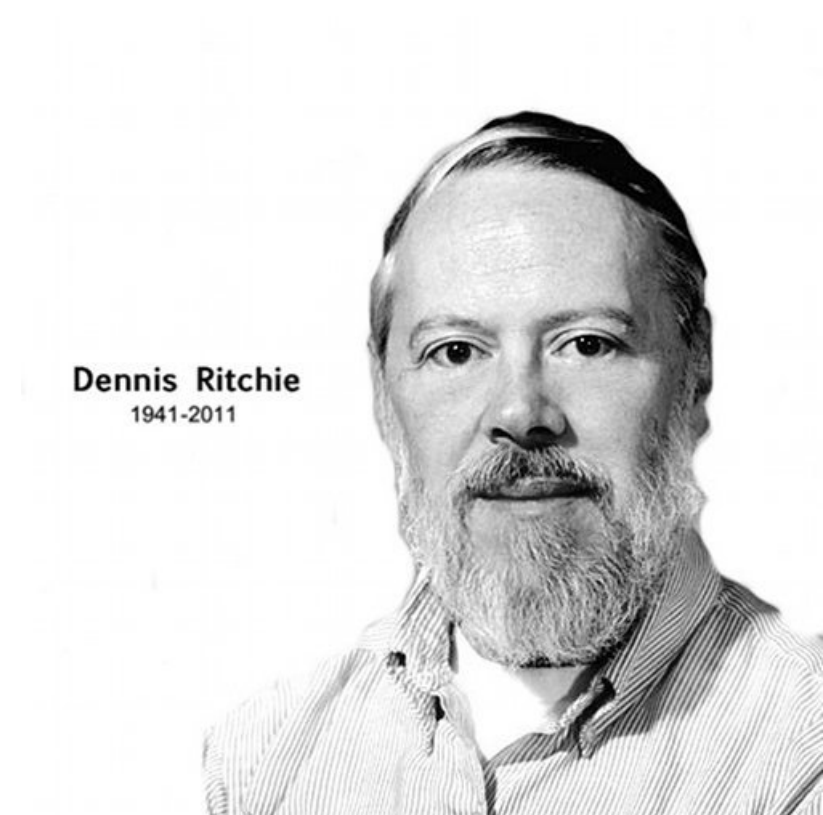
History

Dennis Ritchie (1941-2011)

1970 – Developer for Bell Labs (AT&T)

1972 – C language creation

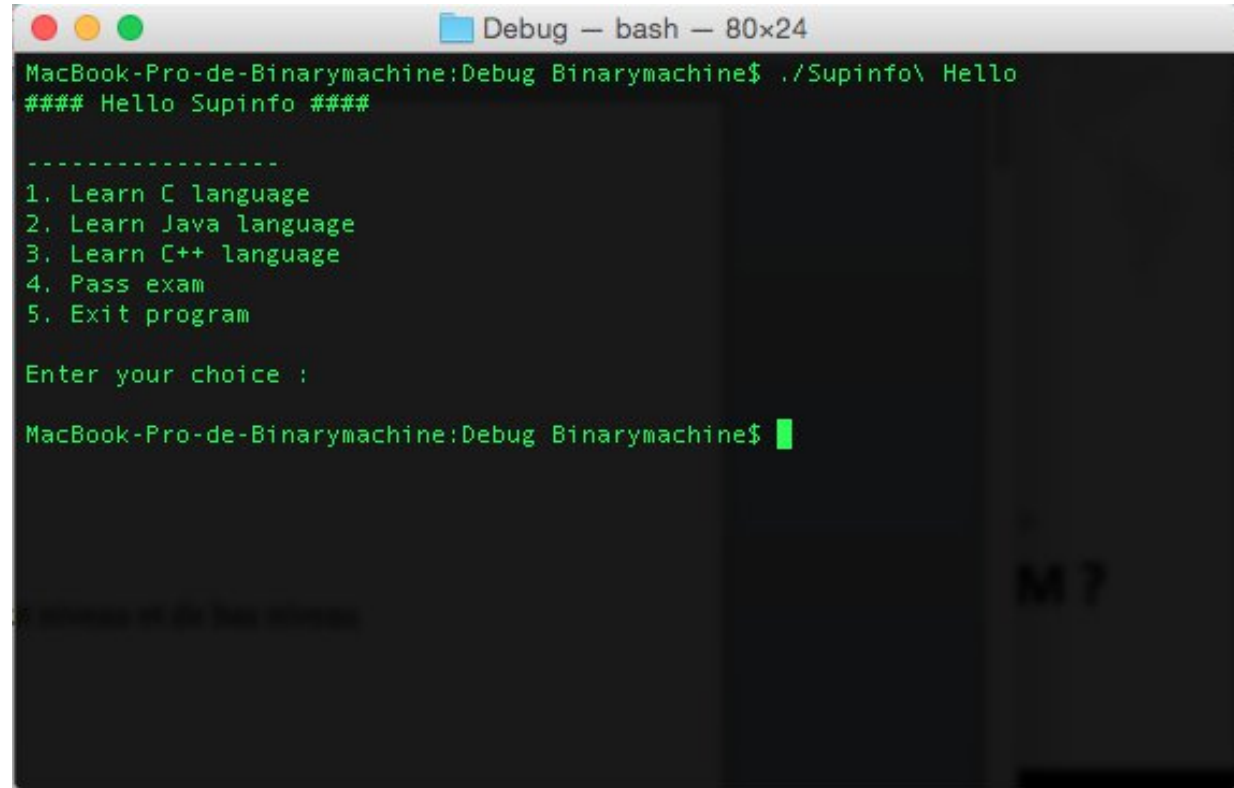
1983 – Turing Award



1. C Language and Programs

Features

- Console application
- Daemon
- Utility
- Driver



```
Debug — bash — 80x24
MacBook-Pro-de-Binarymachine:Debug Binarymachine$ ./Supinfo\ Hello
#### Hello Supinfo ####

-----
1. Learn C language
2. Learn Java language
3. Learn C++ language
4. Pass exam
5. Exit program

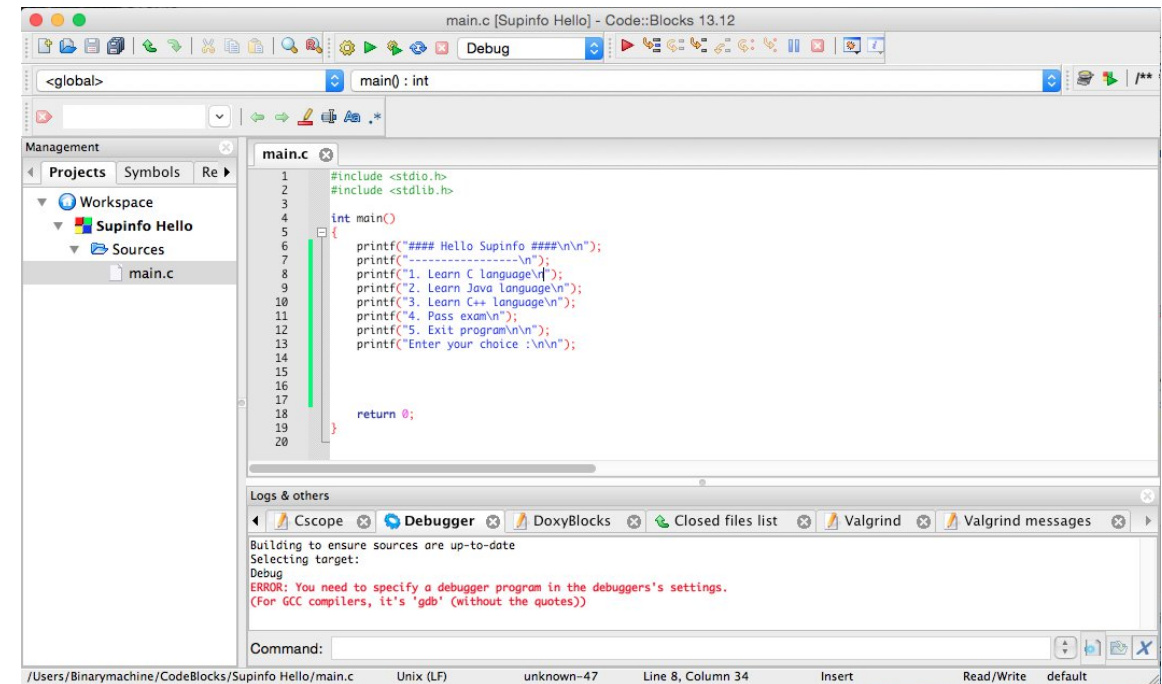
Enter your choice :

MacBook-Pro-de-Binarymachine:Debug Binarymachine$
```


1. C Language and Programs

Features

- Graphic application
- Games

A screenshot of the Code::Blocks IDE. The main window shows 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 Supinfo ###\n\n");
7     printf("-----\n");
8     printf("1. Learn C language\n");
9     printf("2. Learn Java language\n");
10    printf("3. Learn C++ language\n");
11    printf("4. Pass exam\n");
12    printf("5. Exit program\n\n");
13    printf("Enter your choice : \n\n");
14
15
16
17    return 0;
18
19
20 }
```

The 'Logs & others' panel at the bottom shows an error message: 'ERROR: You need to specify a debugger program in the debuggers's settings. (For GCC compilers, it's 'gdb' (without the quotes))'. The status bar at the bottom indicates the file path, compiler, and line/column information.

1. C Language and Programs

Low-level and High-level Languages

- High-level
 - The system management is abstract
- Low-level
 - Memory management can be done by the developer
 - Can manage system resources to optimize

1. C Language and Programs

Low-level and High-level Languages

The C language is used in:

- Operating System
- Embedded System
- Robotics
- Games
- Utility
- Parallel programming
- *etc.*



1. C Language and Programs

Programs

A program is a computer tool that meets a need

- Solution to a need for:
 - Data processing
 - Tasks performing
 - Human/machine interaction
- Each application is:
 - Compiled or Interpreted language
 - Written in understandable language
 - A sequence of instructions

1. C Language and Programs

Questions



2. Tools



2. Tools

Environment

- A text editor
 - To write source code
- A compiler
 - To compile and edit links

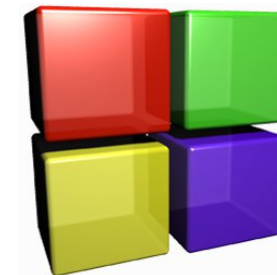
MANDATORY

2. Tools

Environment

IDE examples:

- Visual C++, Dev C++ for Windows
- Xcode for Mac
- Code::Blocks for Windows, Mac and Linux



We will use Code::Blocks for this course!

2. Tools

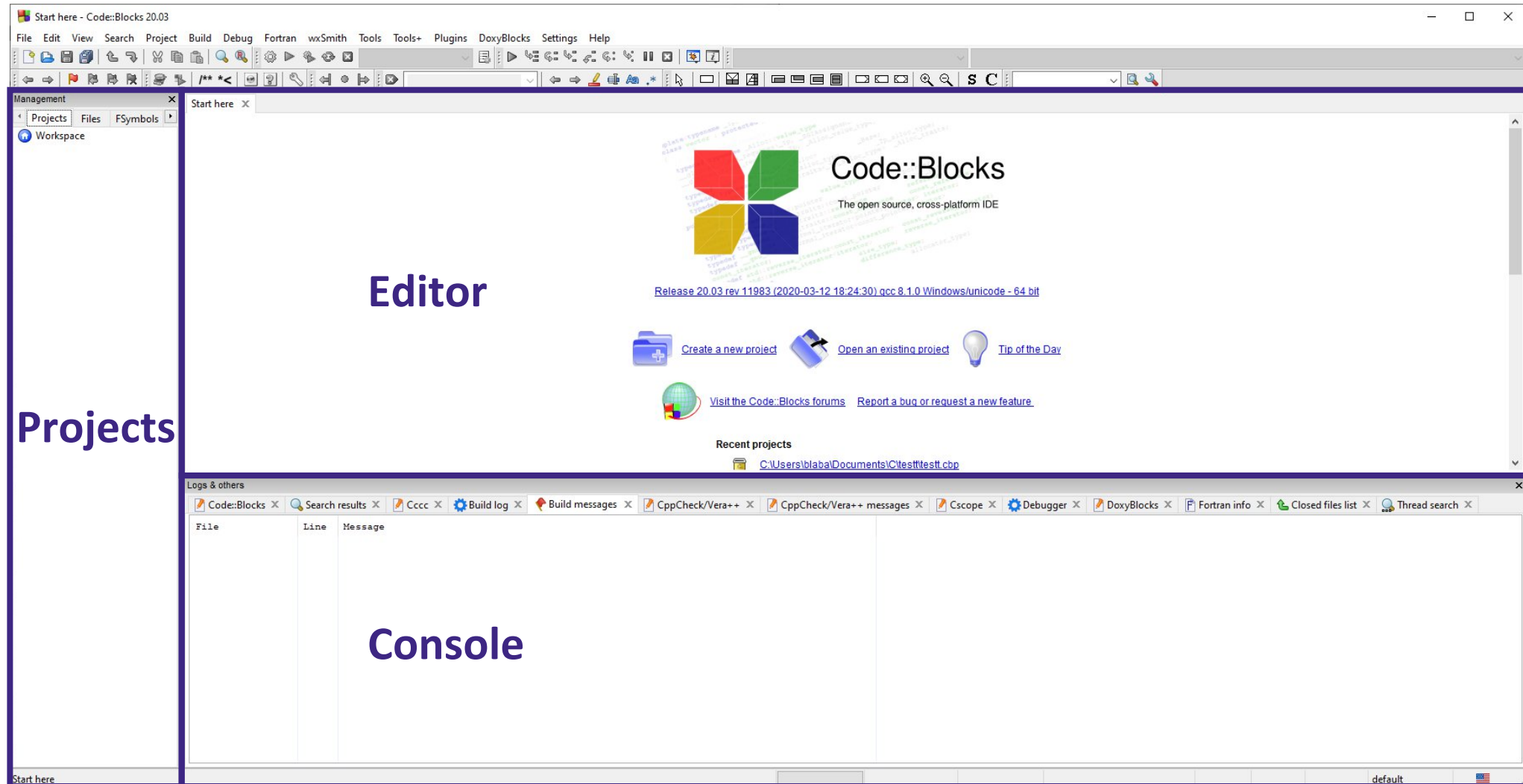
Code::Blocks

- Free
- Open-Source
- Cross-platform
- Designed for C, C++ and Fortran
- Extensible by plugins

<https://www.codeblocks.org/downloads/binaries/>

2. Tools

Code::Blocks



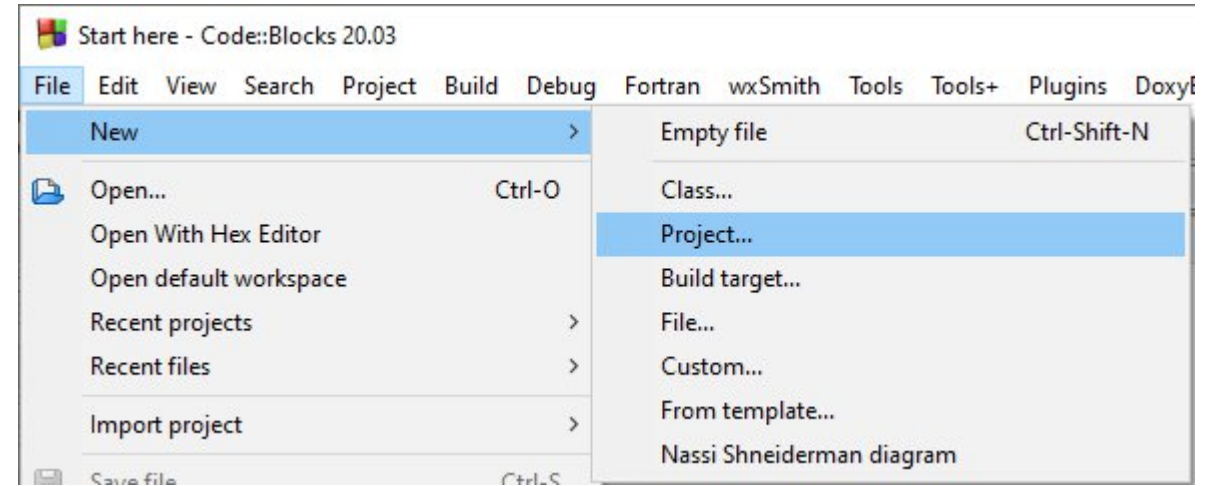
2. Tools

Code::Blocks



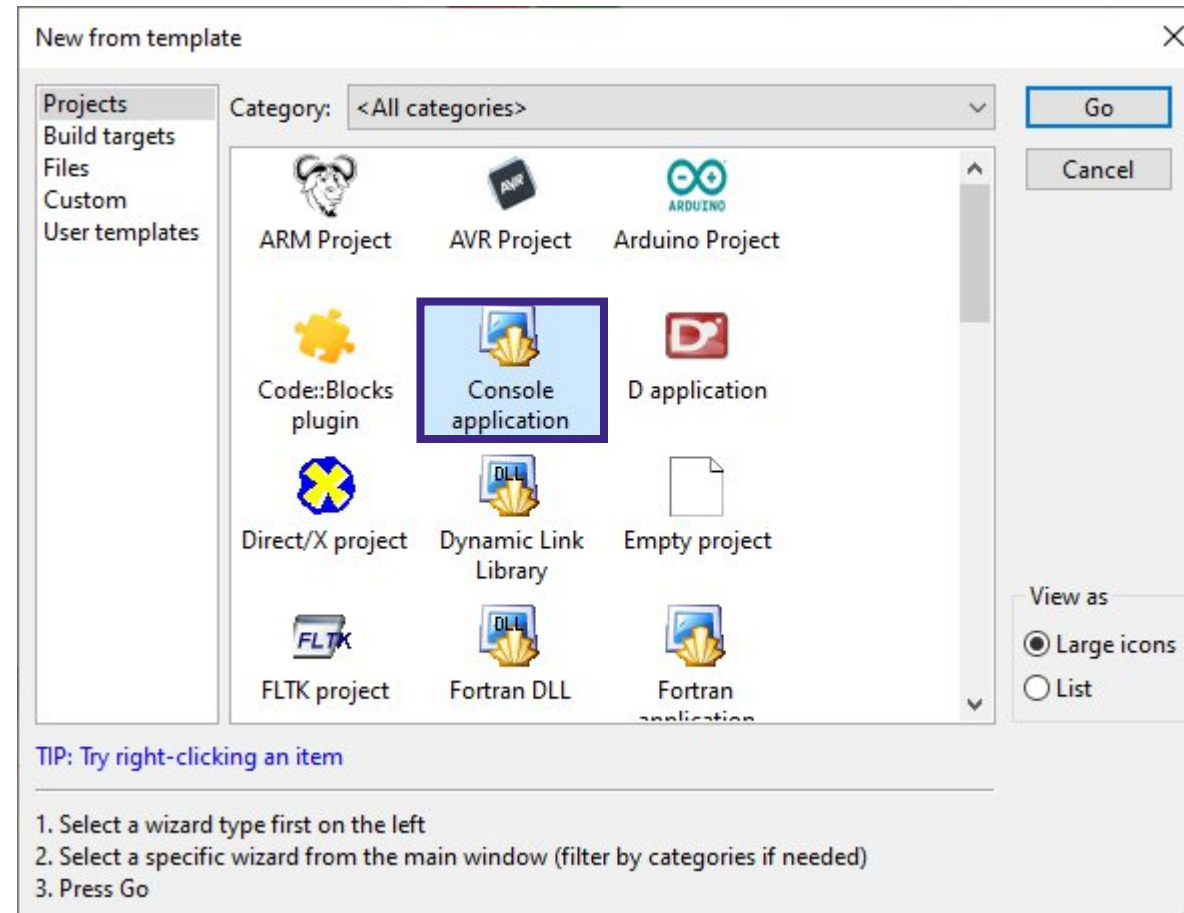
[Create a new project](#)

OR



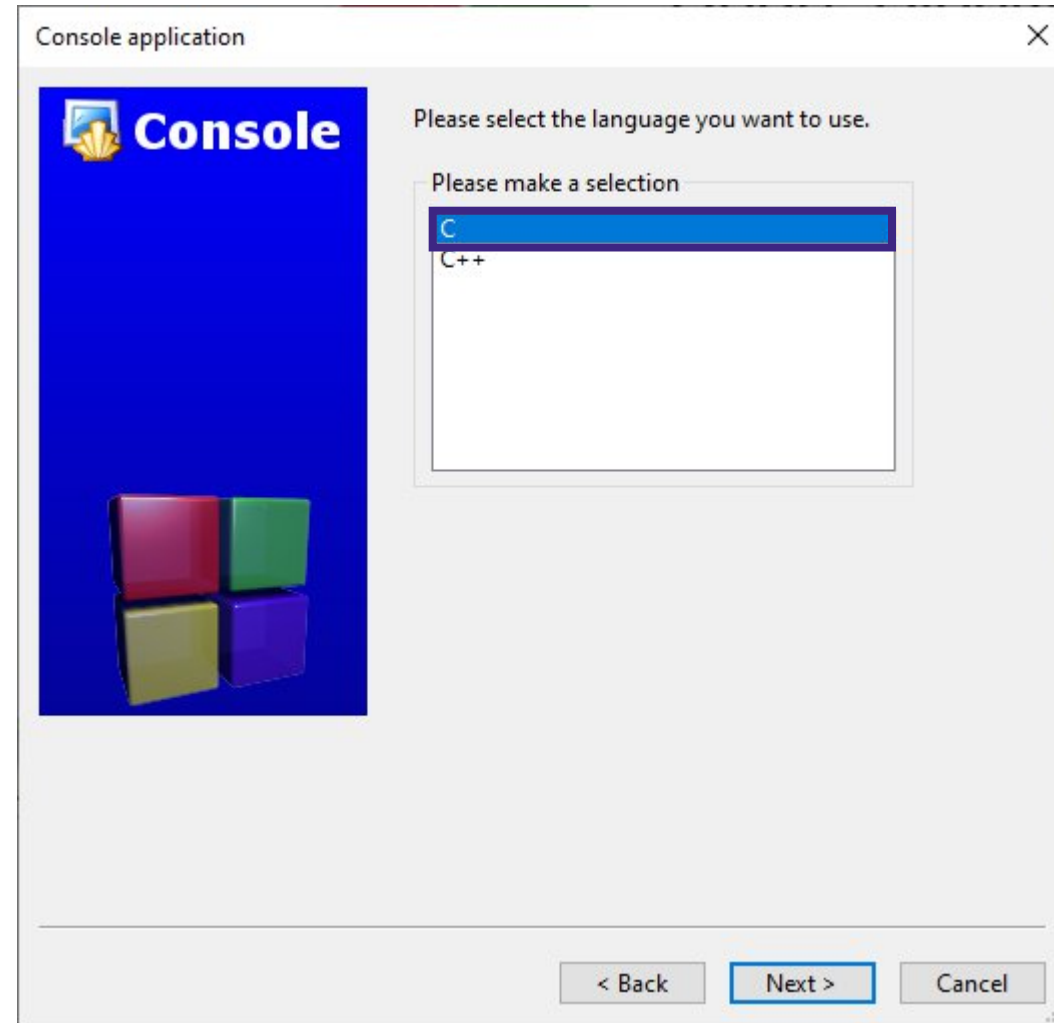
2. Tools

Code::Blocks



2. Tools

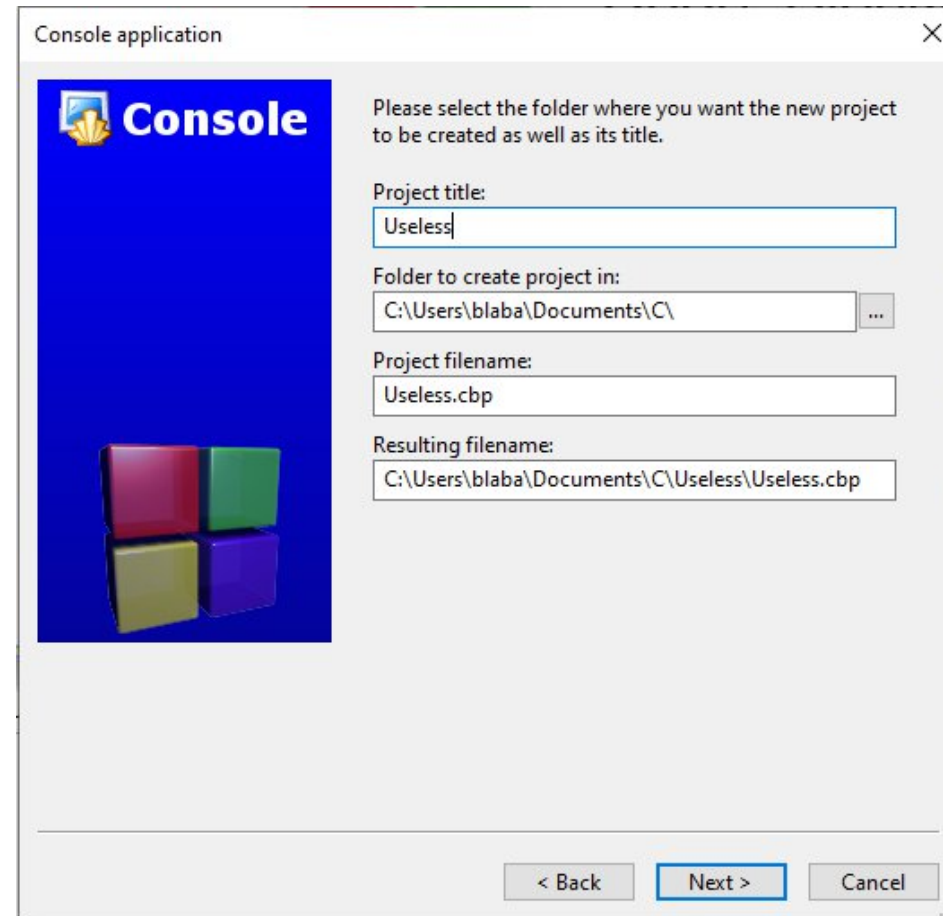
Code::Blocks



2. Tools

Code::Blocks

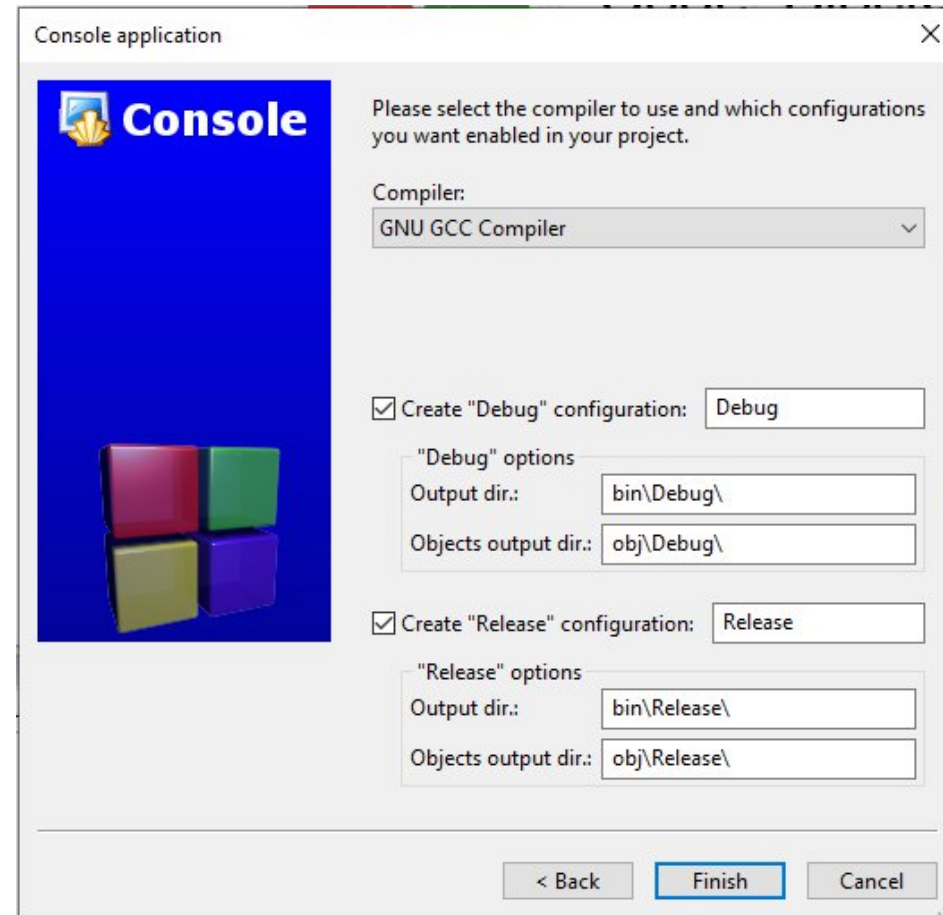
Enter a name for your project and the project location



2. Tools

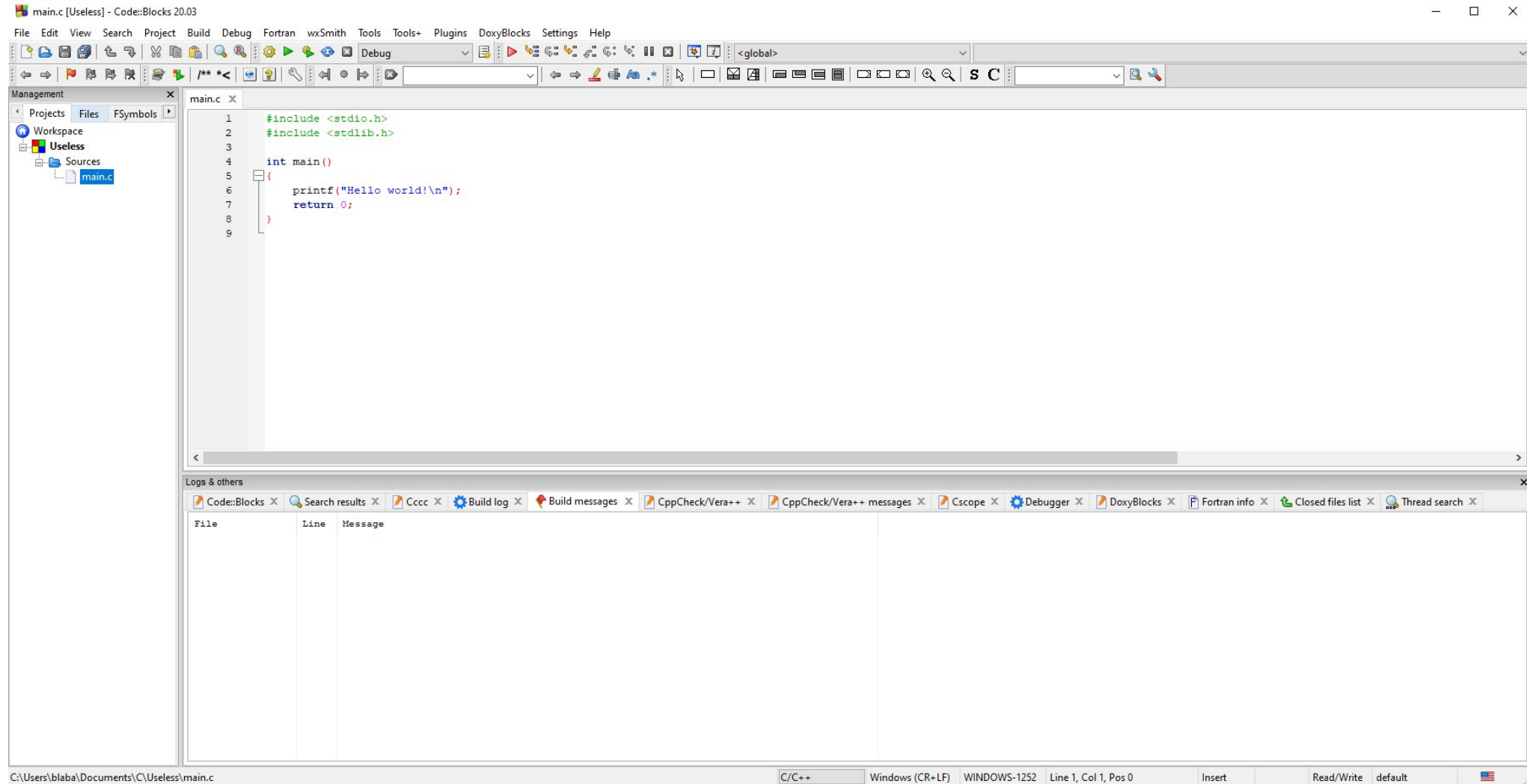
Code::Blocks

Keep the default compilation settings



2. Tools

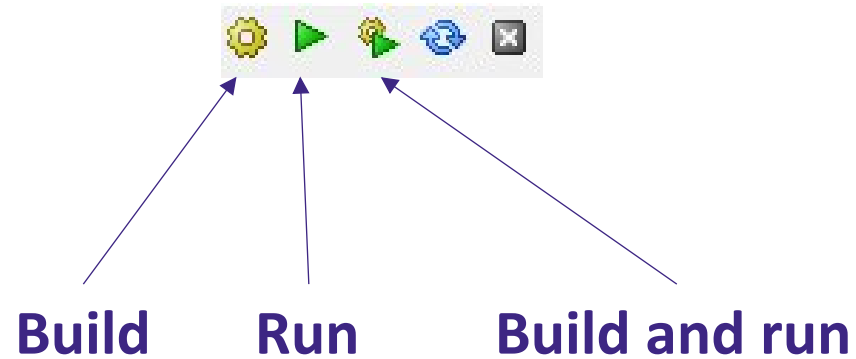
Code::Blocks



2. Tools

Code::Blocks

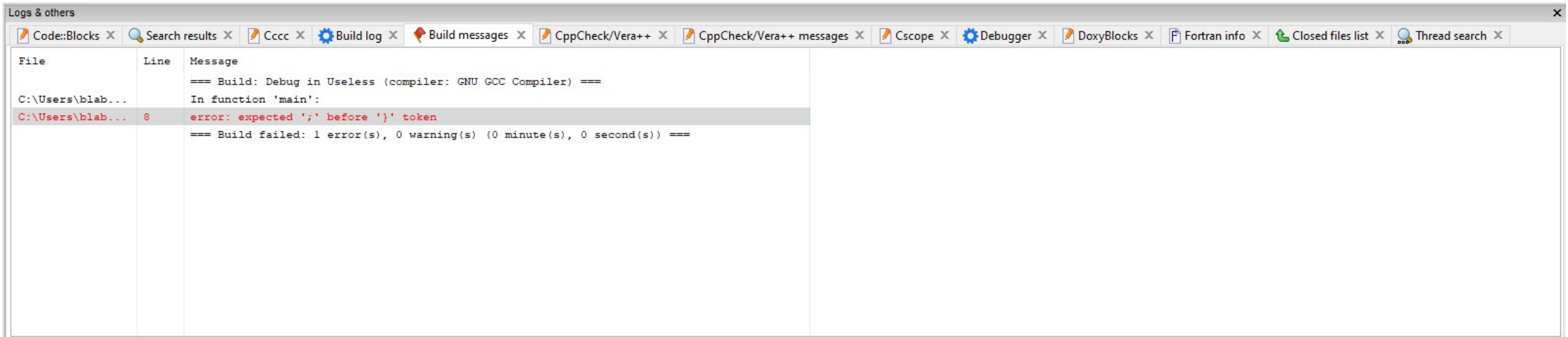
To test a project, click on **Build** then on **Run** or directly on **Build and run**



2. Tools

Code::Blocks

The debugger is a very useful part of the IDE. Find your errors with it.



2. Tools

Questions



3. Program Creation



3. Program Creation

Overview

Three steps:

1. **Write source code** to include the features
2. **Compilation** to convert to binary "machine code"
3. **Link Editing** to group all binary files into an executable file

3. Program Creation

Step 1: Write source code

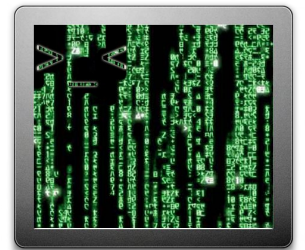
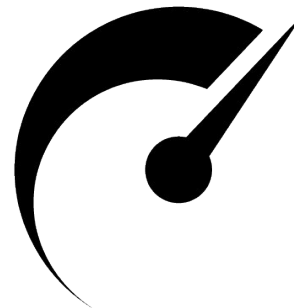
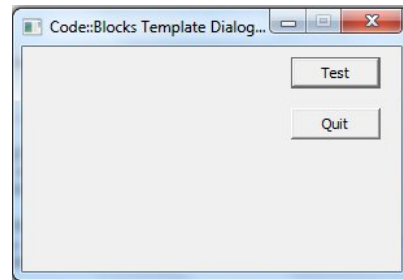
- Source code is a set of text files
- Represent a set of instructions of a program
- May contain libraries
 - **.so** or **.a** files for UNIX
 - **.lib** files for Windows

3. Program Creation

Step 1: Write source code

The C language has a lot of libraries available for:

- Creating games
- Creating GUI applications
- Manipulating environment
- Using specific components
- Saving programming time
- Creating games



3. Program Creation

Step 1: Write source code

All C programs are composed of **.c** files...

```
#if SDL_VIDEO_DRIVER_WINDOWS
extern int SDL_HelperWindowCreate(void);
extern int SDL_HelperWindowDestroy(void);
#endif

/* The initialized subsystems */
#ifdef SDL_MAIN_NEEDED
static SDL_bool SDL_MainIsReady = SDL_FALSE;
#else
static SDL_bool SDL_MainIsReady = SDL_TRUE;
#endif
static SDL_bool SDL_bInMainQuit = SDL_FALSE;
static Uint8 SDL_SubsystemRefCount[ 32 ];

/* Private helper to increment a subsystem's ref counter. */
static void
SDL_PrivateSubsystemRefCountIncr(Uint32 subsystem)
{
    int subsystem_index = SDL_MostSignificantBitIndex32(subsystem);
    SDL_assert(SDL_SubsystemRefCount[subsystem_index] < 255);
    ++SDL_SubsystemRefCount[subsystem_index];
}
```


3. Program Creation

Step 1: Write source code

... and .h files

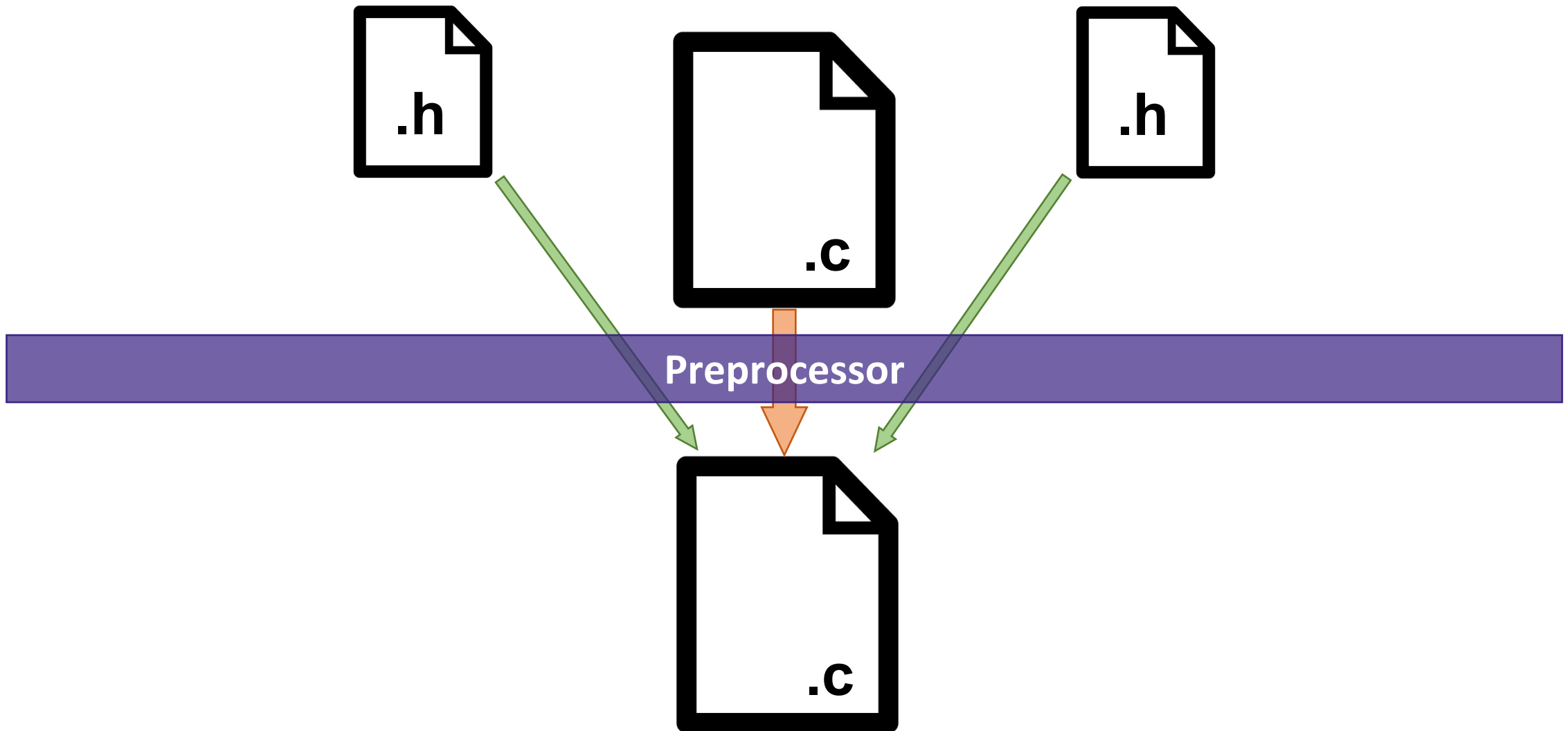
```
#if (defined(__GNUC__) && (__GNUC__ <= 2)) || defined(__CC_ARM)
#define SDL_VARIABLE_LENGTH_ARRAY 1
#else
#define SDL_VARIABLE_LENGTH_ARRAY
#endif

#include "dynapi/SDL_dynapi.h"

#if SDL_DYNAMIC_API
#include "dynapi/SDL_dynapi_overrides.h"
/* force DECLSPEC and SDLCALL off...it's all internal symbols now.
   These will have actual #defines during SDL_dynapi.c only */
#define DECLSPEC
#define SDLCALL
#endif
```

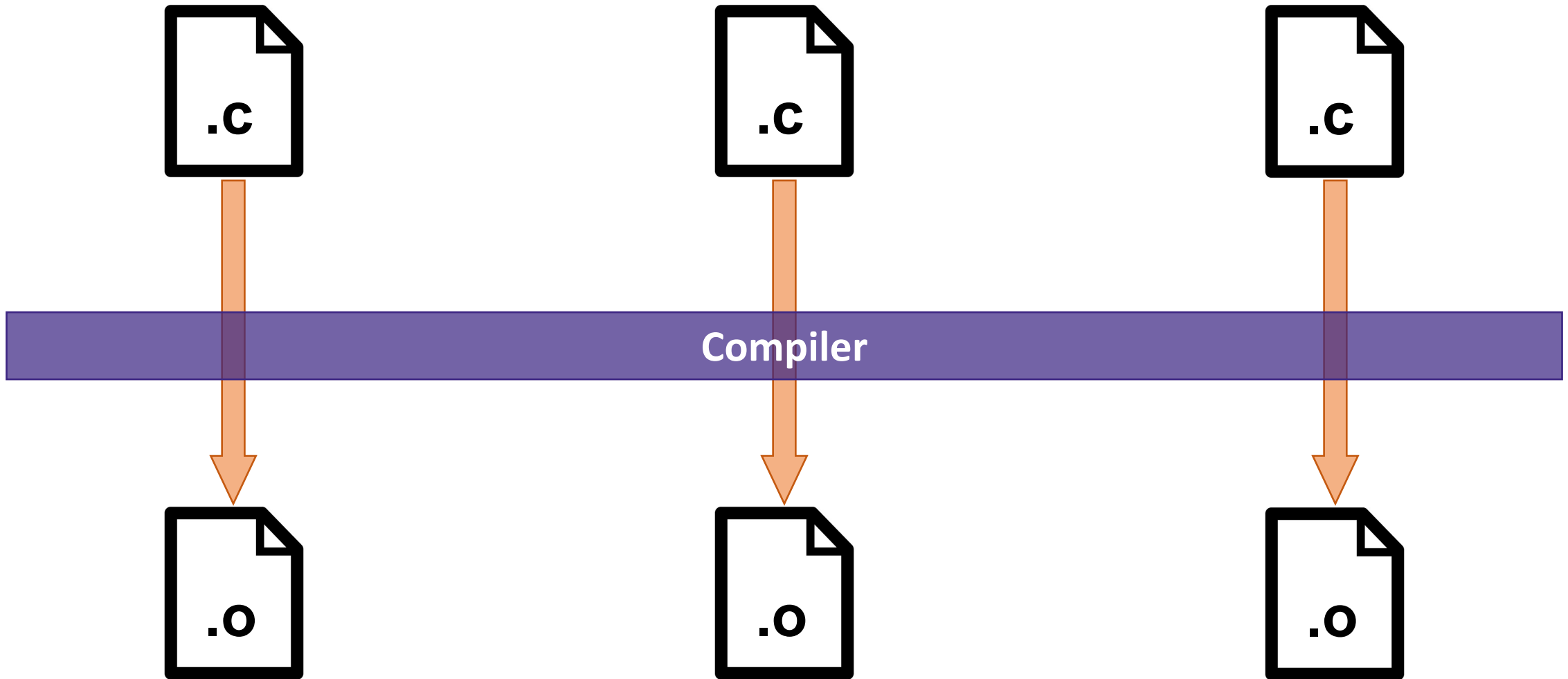
3. Program Creation

Step 2: Compilation



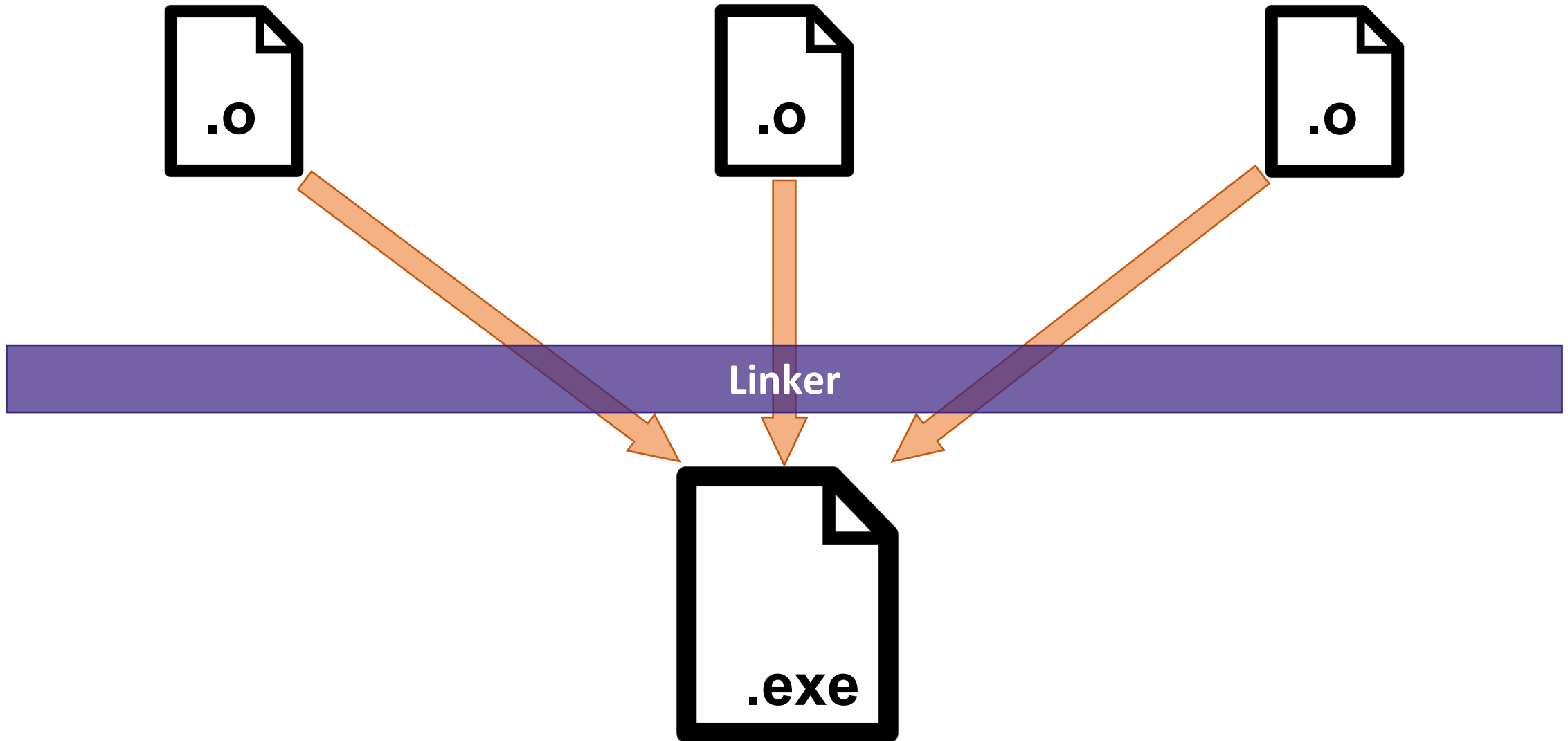
3. Program Creation

Step 2: Compilation



3. Program Creation

Step 3: Link Editing



3. Program Creation

Execute/Run

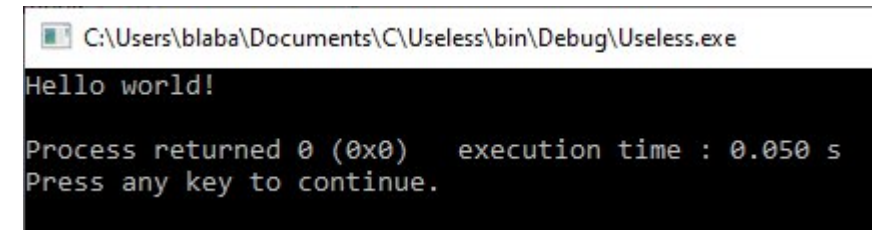
- Linux or Mac

```
./Useless
```

- Windows

```
Useless.exe
```

- Using the IDE **Run** button that opens the console

A screenshot of a Windows command prompt window. The title bar shows the file path: C:\Users\blaba\Documents\C\Useless\bin\Debug\Useless.exe. The command prompt displays the output of the program: 'Hello world!' followed by 'Process returned 0 (0x0) execution time : 0.050 s' and 'Press any key to continue.'

```
C:\Users\blaba\Documents\C\Useless\bin\Debug\Useless.exe
Hello world!
Process returned 0 (0x0) execution time : 0.050 s
Press any key to continue.
```

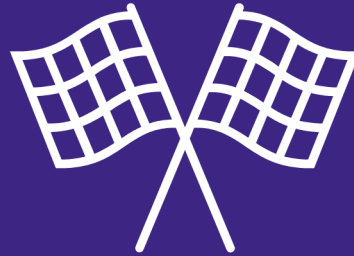
3. Program Creation

Questions



C Developer

Foundation and First Steps



Thank you for your attention