```
$ g++ zero_to_builder.cpp -o builder
```

English edition

C++

C++ · FROM ZERO

ZERO TO BUILDER

The C++ book that explains **every line** and teaches you to **build**, not just memorize.

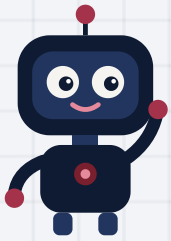
WHAT'S INSIDE

- Every line explained
- 8 real projects
- Quizzes in every chapter
- Exercises at 3 levels
- Memory & pointers made visual
- Final exam + answer key
- Oops corners: real compiler errors
- AI study tips
- Source code + solutions

Elmasryx

Beginner friendly · No experience needed · English + Arabic editions

The language that makes things fast.



BIT · YOUR GUIDE

Hi, I'm **Bit**. I'll be on every page with you. C++ has a reputation for being hard. My job is to make every line clear. Deal?

C++ (say: "see plus plus") is a programming language for building programs that must be **very fast** and use the computer's memory carefully. When a game shows 60 frames every second, or a browser opens a heavy page in a blink, C++ is often working underneath.

Fact	Details
Who made it?	Bjarne Stroustrup, a Danish computer scientist, at Bell Labs.
When?	Work started in 1979 as "C with Classes". Renamed C++ in 1983. First released in 1985.
Why the name?	In C, `++` means "add one". So C++ means "C, one step better".
What kind of language?	**Compiled**: your code is turned into a program file before it runs. (Page 4 explains this.)

Top 4

C++ has stayed in the top 4 of the TIOBE popularity index all through 2026, next to Python, C, and Java.

Source: TIOBE Index, January to August 2026

CHAPTER 00 · WHERE C++ RUNS

You already use C++ every day.

You don’t see C++. You see what it powers. Here are real places where large, important parts are written in C++:

Game engines
Unreal Engine is written mainly in C++. Many big games are built with it.

Web browsers
Chrome and **Firefox** use C++ for the core that draws web pages.

Operating systems
Large parts of **Windows** are written in C and C++.

Creative apps
Adobe Photoshop uses C++ for its heavy image work.

Databases
MySQL is written in C and C++ to handle huge amounts of data.

Small devices
Arduino boards, robots, and car systems often run C or C++.

Why do they choose C++?

Speed
It becomes machine code, so it runs very close to the hardware.

Control
You decide how and when memory is used.

Lasts long
C++ code written years ago still compiles today.

Big systems
Classes help organize programs with millions of lines.

Elmasryx

What you can do with C++, and what you can't.

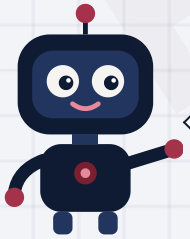
✅ With C++ you can...

- Build **games** and game engines (Unreal Engine uses C++).
- Write programs that need **top speed**: video, 3D, simulations.
- Program **hardware**: Arduino, robots, sensors.
- Do **competitive programming** (most contestants use C++).
- Understand **how computers really work**: memory, addresses.

⚠️ Without C++ (or not the best tool)...

- **Websites**: browsers run JavaScript, not C++.
- **Quick scripts and AI experiments**: Python is faster to write.
- **Phone apps**: Android uses Kotlin, iPhone uses Swift.
- **Simple office tasks**: C++ is too much work for small jobs.
- Learning it is slower: you control more, so you write more.

	Python	JavaScript	C++
Speed when running	Slower	Fast	**Very fast**
Easy for beginners	**Easiest**	Easy	Harder
How it runs	Interpreted	Interpreted / JIT	**Compiled** to a program file
You manage memory?	No	No	**Yes, you can**
Best for	AI, data, automation	Websites, web apps	Games, systems, speed



BIT'S TIP

Learning C++ makes **every other language easier**. After it, Python and JavaScript feel like a holiday.

CHAPTER 00 · HOW C++ WORKS

Your code becomes a real program.

Python and JavaScript read your code **line by line while it runs**. C++ is different: a tool called the **compiler** reads your **whole** code first, checks it, and turns it into a program file that the computer runs directly.

`hello.cpp`
1. You write
Your code in a text file. The ending is `.cpp`.

→

`g++`
2. Compiler
Checks every line, then translates it to machine code.

→

`hello.exe`
3. Program file
A real app. On Mac/Linux it's just `hello`.

→

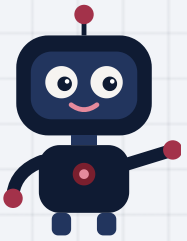
`▶ run`
4. It runs
The computer runs it directly. Very fast.

Compile time vs run time

	Compile time	Run time
When?	When you build the program (step 2)	When the program is running (step 4)
Example error	A missing `;` → the compiler refuses	Dividing by zero → the program crashes
Good news	Many mistakes are caught **before** anyone uses your app	Fewer surprises, because the compiler checked first

> **REMEMBER**
> If you change your code, you must **compile again**. The old program file does not update by itself.

BIT
Think of a translator who translates a **whole book** before printing it. Slower to start, but the printed book reads fast. That's C++.



CHAPTER 00 · SETUP

Ready in 15 minutes.

You need two free things: an **editor** to write code (VS Code), and a **compiler** to build it (g++). Pick your system:

1 Install VS Code

Download it free from **code.visualstudio.com**. Open it, go to **Extensions**, and install **C/C++** by Microsoft.

2 Install the compiler

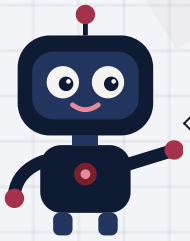
Windows: follow the official VS Code guide "Using GCC with MinGW". It installs g++ for you.
Mac: open Terminal and type `xcode-select --install`
Linux: type `sudo apt install g++`

3 Check it works

Open a new terminal in VS Code (**Terminal → New Terminal**) and type:

```
TERMINAL
$ g++ --version
g++ (GCC) 13.3.0
```

You see a version number? You're ready. (Your number may be different. That's fine.)



BIT'S TIP

Windows: after installing, **close and reopen VS Code**. See `g++ is not recognized`? The compiler isn't in PATH yet; the guide shows the fix. No laptop? Search "online C++ compiler" and use your phone.

CHAPTER 00 · YOUR FIRST PROGRAM

Five lines. Every one explained.

Create a file called `hello.cpp` and type this exactly:

HELLO.CPP

```cpp
#include <iostream>

int main() {
    std::cout << "Hello, world!" << "\n";
    return 0;
}
```

LINE BY LINE

`#include <iostream>`	Bring in the **input/output** tools. Without it, `std::cout` doesn't exist.
`int main() {`	Every C++ program starts in `main`. `int` means it gives back a whole number. `{` opens its body.
`std::cout << "Hello, world!"`	`std::cout` = the screen. `<<` = "send this to it". `std::` means "from the standard library".
`<< "\n";`	`\n` = new line. The `;` ends the instruction, like a full stop.
`return 0 ;` `}`	Return 0 = "finished with no problems". `}` closes `main`.

TERMINAL

```
$ g++ hello.cpp -o hello
$ ./hello
Hello, world!
```

`-o hello` names the program file. On Windows run `hello.exe` or `.\hello`.



OOPS CORNER

Forgot a `;`? The compiler stops and tells you the line:

```
error: expected ';' before 'return'
```

Fix: add the `;` at the end of the line above. Then compile again.

01

CHAPTER 01

Variables & Types

A variable is a **box with a name** that holds a value. In C++, every box also has a **type**: what kind of value it can hold. You must say the type when you create the box.

WHAT IT DOES

Stores values so your program can remember and use them.

WHY YOU NEED IT

C++ uses the type to know **how much memory** to reserve and what you're allowed to do with the value.

TYPES.CPP

```cpp
#include <iostream>
#include <string>
using namespace std;

int main() {
    int age = 24;
    double price = 9.99;
    char grade = 'A';
    bool isMember = true;
    string name = "Sara";

    cout << name << " is " << age << ", grade " << grade << "\n";
    cout << "Price: " << price << ", member: " << isMember << "\n";
    return 0;
}
```

OUTPUT

```
Sara is 24, grade A
Price: 9.99, member: 1
```

Type	Holds	Example	Note
`int`	Whole numbers	`24`, `-5`	No decimals
`double`	Decimal numbers	`9.99`	Use for money and math
`char`	One character	`'A'`	Single quotes `' '`
`bool`	true / false	`true`	Prints as 1 or 0
`string`	Text	`"Sara"`	Double quotes, needs `<string>`