$ python3 zero_to_builder.py

English edition

PYTHON

ZERO TO BUILDER

The Python book that teaches you to **build**, not just memorize.

WHAT'S INSIDE

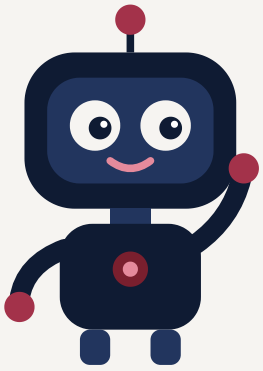
- Step-by-step explanations
- 8 real projects
- Quizzes in every chapter
- Exercises at 3 levels
- Final exam
- Answer key + solution code
- Mistake Museum
- AI study tips
- Source code included

Elmasryx

Beginner friendly · No experience needed · English + Arabic editions

CHAPTER 00 · START HERE

Hey! Let's turn you into a builder.



BIT · YOUR GUIDE

Hi, I'm **Bit**. I'll be on every page with you. When something looks scary, I'll jump in and make it simple. Deal?

Python is the language behind AI tools, the data companies run on, and the tiny scripts that do **3 hours** of boring work in **3 seconds**. Get good at it, and it pays you back:

Get hired
AI, data, and automation jobs use it daily.

Freelance
Clients pay for scripts that save them hours.

Level up your job
Automate the boring parts. Get noticed.

Build your own
Turn a script into a tool you can sell.

#1 Python is the most used language by people learning to code. 71.8% of them use it.

Source: Stack Overflow Developer Survey 2025

Elmasryx

CHAPTER 00 · SETUP

Ready in 10 minutes.

You need two free things: **Python**, the language, and **VS Code**, the notebook where you write it. That's the whole shopping list.

1

Get Python

Go to **python.org/downloads**, download it, and click through the installer.

2

Get VS Code

Download it from **code.visualstudio.com**. Open it, click **Extensions**, search **Python**, install the one by Microsoft.

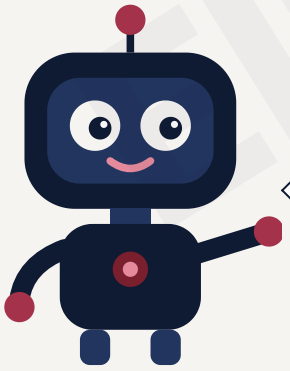
3

Say hi to Python

Open **Terminal → New Terminal** and type:

```
python --version    # Windows
python3 --version   # Mac / Linux
```

See something like `Python 3.13`? You're in.



BIT'S TIP

Windows friends: when installing, tick **"Add python.exe to PATH."** Skip it and you'll be googling errors at midnight. Ask me how I know.

NO LAPTOP RIGHT NOW?

Search "online Python editor." You can run every example in this book from your phone's browser.

One line. Your first program.

Create a file called `hello.py` in VS Code. The `.py` at the end tells the computer "this is Python." Now type this:

HELLO.PY

```
print(1)("Hello, world!"(2))
```

1. `print` is a command. It means: **"show this on the screen."**
2. Words go inside **quotes**. That's how Python knows it's text, not a command.

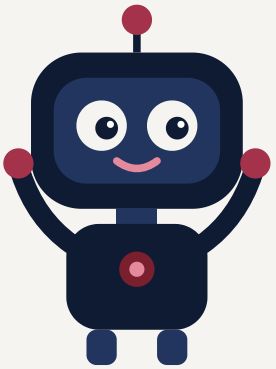
Click the ▶ button at the top right. And...

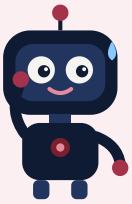
OUTPUT

```
Hello, world!
```

BIT

You just told a computer what to do, **and it listened.** Every developer on the planet started with this exact line. Welcome to the club.





OOPS CORNER

Forgot the quotes? Python thinks `Hello` is a name it should know:

```
NameError: name 'Hello' is not defined
```

Fix: put your text back inside quotes.

CHAPTER 01

01 Python Basics

Every app on your phone does three things. After this chapter, your programs will too:

Show
Put things on the screen

Remember
Keep info for later

Listen
Take answers from the user

`print` Show anything, even math

What it does Shows text, numbers, or results on the screen.

Why you need it It's how your program talks back to you, and how you check that your code works.

BASICS.PY

```
# Python skips this line. It's a note for you. (1)
print("I'm learning Python!")
print("2 + 3 =", (2) 2 + 3) (3)
```

1. A line that starts with `#` is a **comment**: a sticky note for humans.
2. A **comma** lets you show several things on one line.
3. No quotes around math? Python **calculates** it for you.

OUTPUT

```
I'm learning Python!
2 + 3 = 5
```

YOUR TURN

How many minutes are in a week? Don't grab a calculator. Make Python do it: `print(7 * 24 * 60)`

REMEMBER THINGS

A variable is a box with a label.



BIT

The label on the box is the **name**. What you put inside is the **value**. Say the name later, and Python opens the box for you.

PROFILE.PY

```
name = "Sara"
age = 24
age = age + 1      # happy birthday!
print(name, "is now", age)
```

1. The label. No spaces allowed, so use `user_name`, not `user name`.
2. `=` means **“put this inside.”** It’s not “equals” like in math class.
3. The value. You can swap it anytime, like we did with the birthday.

OUTPUT

```
Sara is now 25
```

4 kinds of things a box can hold

"Sara"	24	1.68	True
`str`	`int`	`float`	`bool`
Text. Always in quotes.	Whole numbers.	Decimal numbers.	Yes or no. True or False.

LISTEN TO THE USER

Let your program have a conversation.

YOUR PROGRAM
What's your name?

THE USER
Omar

YOUR PROGRAM
How old are you?

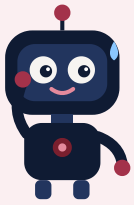
THE USER
21

YOUR PROGRAM
Hi Omar! In 10 years you'll be 31.

FUTURE.PY

```
name = input(1)("What's your name? ")
age = int(2)(input("How old are you? "))
print(f"Hi {name}! In 10 years you'll be {age + 10}.")(3)
```

1. `input` asks a question and **waits** for the user to type.
2. Answers always arrive as **text**. `int()` turns text into a number so you can do math.
3. Put `f` before the quotes, then drop any variable inside `{ }`. Python fills in the blanks.



OOPS CORNER

Text and numbers don't mix. `"5" + "5"` gives you `55`, not `10`. Python glues text together, it doesn't add it. Forget `int()` and you'll meet this:

```
TypeError: can only concatenate str (not "int") to str
```