CODAPPS

Essential notions of code

Clément Levallois

2018-01-22

Table of Contents

last modified: 2018-02-02

1. The interactive tool we are using to learn the basics of code

It is useful to learn about coding by reading a lesson like this one, but it is more efficient if you can **practice while you read**.

Indeed, a key competency in coding is to learn the simple discipline of writing text carefully:

- not forgetting a `;` at the end of the line
- not forgetting to put a capitalized letter when it is necessary
- learning how to spot opening accolades `{` and closing accolades `}`
- not confusing when to use commas `,` and semi-colons `;`

It is all very trivial, right? And yet this is what needs the most practice, at the start.

So in this lesson, you will see plenty of interactive screens like this one:

You can write your code directly on it in this lesson, and and "launch it" to see the results. Here is an explanation of how this works:

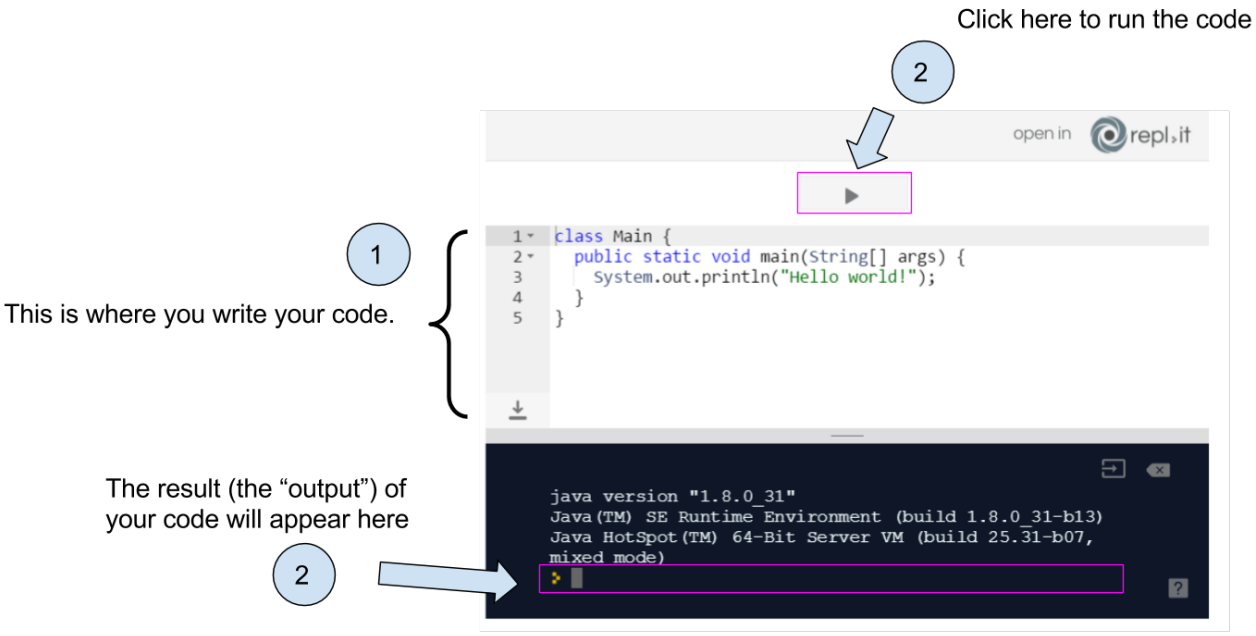

Figure 1. How does the interactive tool work

Last note before we start: this lesson is on the essentials of coding, so we go fast and discuss just the most important notions of coding.

If you are interested in (much) longer, more thorough approaches, you can have a look at these two interactive courses:

- "Think Java" by Tinklet (Java is the same programmming language as the one we use here)
- "Python for Everybody" also by Tinklet (Python is the most popular programming language for data science)

1. Variables

If we compare coding to cooking, you could say that `variables` are the ingredients of the recipe.

If we develop a gaming app, we need to have characters and animate them, show their names on screen, record scores...

→ variables are a way to create, define and store all these things we need in our program.

You can create all the kinds of variables you want, but to help you, a number of them are already predefined because they are so common:

A variable is created and defined this way:

Creating a variable

```
String playerName = "Bernard B";
```

What does all this mean? Let's decompose each term:

- `String`: The capital **S** is mandatory. It means this variable specializes in storing **text**. This is the **type** of the variable.
- `player`: this is the name I chose for the variable, it could have been anything else. By convention it always starts **without** a capitalized letter and it has **no space** in it.
- `=`: the right of the equal sign will be the value of the variable.
- `"Bernard B"`: this is the value I store in the variable `player`. **Textual values should be put between quotes " ".**
- `;`: this step of code is finished a the end of the line. The `;` shows this end and **is mandatory** (if you don't put it, the program tries to read the next line as the direct continuation of this one, and it gets confused).

Creating a variable

```
String playerName = "Bernard B";
```

So this single line of code creates a variable called playerName, and I immediately give it a value: "Bernard B".

This will be handy when we need to show the player's name on the screen of the app: we will just use the variable `player`, and what ever value in it (the player's name) will be shown.

Try to create a variable by yourself:

You need to put the name of your variable in the line "System.out.println..." if you want to see the result on screen, just like below

The end

Questions? Want to open a discussion on this lesson? Visit the forum here (need a free Github account).

Find references for this lesson, and other lessons, here.

This course is designed by Clement Levallois.

Discover my other courses in data / tech for business: http://www.clementlevallois.net

Or get in touch via Twitter: @seinecle