Expressions and Statements

Let's write our first piece of code in Reason and understand how it works.

In ReasonML, almost every line of code is an **expression**. An expression returns a certain value. It can be anything from a simple mathematical equation to a complex function. We don't need to worry about them for now.

Apart from expressions, **statements** are also part of Reason. Statements usually print something to the *console* instead of returning a useful value.

The "Hello World" Statement

With the definitions out of the way, we can move on to writing our first piece of code. We'll follow tradition and begin by printing "Hello World" to the console:

```
Js.log("Hello World");
```

The similarity to JavaScript syntax is undeniable. Instead of JavaScript's `console.log()`, Reason uses `Js.log()` to print the statement enclosed within. The `"Hello World"` text is just a string which is being printed.

Strings will come up later in the course.

The Semi-Colon

In ReasonML, every expression and statement, no matter how short or big, ends with a semi-colon. Without getting into the detail of the code, just have a look at how the semi-colons appear in the following code:

```
Js.log("Educative"); /* Semi-Colon at the end of a statement */

type circle = {
  radius: int,
  color: string
}; /* Semi-Colon after a block */

5 + 5; // Semi-Colon after a mathematical expression
```

Comments are placed within the `/*` and `*/` symbols. Single line comments can also be written after `//`. Comments are used solely for documentation and play no role in the actual code execution.

By now, we're familiar with the simplest pieces of code in ReasonML.

The next lesson is a quiz to help reinforce what you have learned so far.