Operator Functions

Let's learn how to create our own operators in ReasonML.

WE'LL COVER THE FOLLOWING

In Reason, we can use functions to create our own customized operators. The purpose of the operator will be defined within the function.

All the predefined in-fix operators can be used to create new in-fix operators. The same goes for prefix operators.

The Logic

Let's go back to operators for a bit. An operator is used with a value or values to produce an output. If we consider the operator symbol as the *function name* and the value(s) as *arguments*, an operator follows the conventions of a function.

In fact, operators *are* functions. This implies that we can create our own operators just like we can create our own functions.

The Template

Below, we can see that the template for creating operators is almost identical to that of functions:

```
let (operator) = (arguments) => expression;
```

It is important to enclose the operator in parentheses.

The Syntax

Let’s create the `*+` infix operator which converts an integer to a string and concatenates it to another string.

```
let (*+) = (str: string, n: int) => {
    str ++ string_of_int(n);
};

let x = "Educative " *+ 2019;
Js.log(x);
```

For the second example, we’ll create the `~~` pre-fix operator which computes the length of a string.

```
let (~~) = (str) => String.length(str);
let lower = "Hello";
Js.log(~~lower);
```

In the next lesson, we’ll create **polymorphic functions**.