Introduction

In this section, we'll learn why the use of functions is essential in ReasonML.

WE'LL COVER THE FOLLOWING

What is a Function?

The concept of functions is one of the most powerful tools in modern programming.

A function, or **method**, can be thought of as an entity that contains a set of operations. Whenever a function is *called* the operations are executed in order to return a value. In this sense, functions themselves can be thought of as predefined expressions.

A big syntactical advantage of functions is that they erase the need to write the same code over and over again. A function acts as a block which performs an operation we need to reuse over and over again.

Reason's Approach

Reason is a **functional programming language** which means that it relies heavily on functions. Functions are responsible for the flow of information in a Reason application.

Examples

We've already seen several instances of the built-in functions of Reason. For arrays, we have the `Arrays.make()` function to create a dynamic array. The `String.contains()` method was also an example of a function. Both of these

functions performed a predefined task. We did not have to write the logic behind them again and again in our program. That is the beauty of functions.

Though it may feel a bit weird that we are discussing them so late in the course, do not worry. The concepts which we've studied up to now will only make functions easier to understand.

In the following lessons, we'll understand the true potential of functions in Reason.

Let's get started by creating our first function!