Discovering Functions

This lesson discusses the most important component of any programming language i.e., functions. We will go through all the basic concepts of functions in this lesson.

We'll cover the following

The Role of Functions

To understand why functions are important, check out our example from a previous chapter: the burrito algorithm :)

```
Begin
  Get out the rice cooker
  Fill it with rice
  Fill it with water
  Cook the rice
  Chop the vegetables
  Stir-fry the vegetables
  Taste-test the vegetables
    If the veggies are good
      Remove them from the stove
    If the veggies aren't good
      Add more pepper and spices
    If the veggies aren't cooked enough
      Keep stir-frying the veggies
  Heat the tortilla
  Add rice to the tortilla
  Add vegetables to the tortilla
  Roll tortilla
End
```

Here's the same general idea, written in a different way.

```
Begin
  Cook rice
  Stir-fry vegetables
  Add fillings
  Roll together
End
```

The first version details all the individual actions that make up the cooking process. The second breaks down the recipe into *broader steps* and introduces concepts that could be re-used for other dishes as well like *cook*, *stir-fry*, *add* and *roll*.

Our programs so far have mimicked the first example, but it's time to start modularizing our code into sub-steps so we can re-use bits and pieces as needed. In JavaScript, these sub-steps are called *functions*!

Discovering Functions

A *function* is a group of statements that performs a particular task.

Here's a basic example of a function.

```
function sayHello() {
  console.log("Hello!");
}

console.log("Start of program");
sayHello();
console.log("End of program");
```

Let's study what just happened.

Declaring a Function

Check out the first lines of the example above.

```
function sayHello() {
  console.log("Hello!");

}
```

This creates a function called `sayHello()`. It consists of only one statement that will make a message appear in the console: `"Hello!"`. This is an example of a function *declaration*.

```
// Declare a function called myFunction
function myFunction() {
  // Function code
}
```

The declaration of a function is performed using the JavaScript keyword `function`, followed by the function name and a pair of parentheses. Statements that make up the function constitute the *body* of the function. These statements are enclosed in curly braces and indented.

Calling a Function

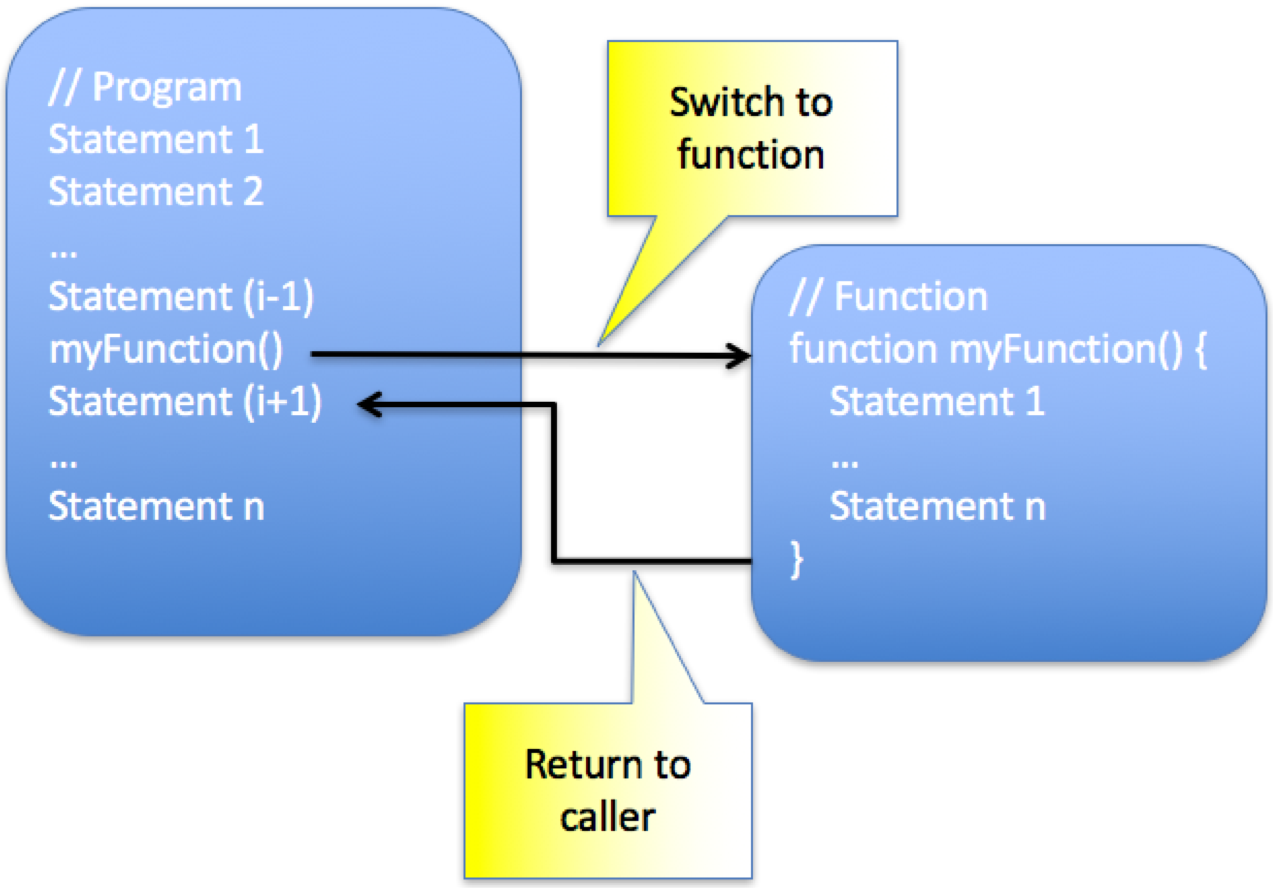
Functions must be called in order to actually run. Here's the second part of our example program.

```
console.log("Start of program");
sayHello();
console.log("End of program");
```

The first and third statements explicitly display messages in the console. The second line makes a *call* to the function `sayHello()`. You can call a function by writing the name of the function followed by a pair of parentheses.

```
// ...
myFunction(); // Call myFunction
// ...
```

Calling a function triggers the execution of actions listed therein (the code in its body). After it's done, execution resumes at the place where the call was made.



Usefulness of Functions

A complex problem is generally more manageable when broken down into simpler subproblems. Computer programs are no exception to this rule. Written as a combination of several short and focused functions, a program will be easier to understand and to update than a monolithic one. As an added bonus, some functions could be reused in other programs!

Creating functions can also be a solution to the problem of code duplication; instead of being duplicated in several places, a piece of code can be centralized in a function and called from anywhere when needed.