Coding Basics Badge

What Are Functions?

Functions are a common type of instruction in programming that tell a computer to perform a certain task.

For example, look at this function:

drawNose();

This function would tell a computer to draw a nose — perhaps on a web site page.

In coding, using a function is *calling* a function, or *invoking* a function.

In order for a function to be called, it must be previously *defined*, or *declared*. That means that once you have created a function, or defined, a function, you can use it, or call it, again and again in your code. The rules for writing code are called **syntax**.

For example, to call a function with Javascript, you would write it like this:

theNameOfTheFunction();

Here are the rules for calling (or writing) a function in JavaScript:

- The function starts with a name, which can't have any spaces or use other special characters (except underscores "_").
- The name can use numbers, but not at the beginning.
- The name is directly followed by parentheses "()". The "()" is what tells JavaScript to run the function.
- The semicolon at the end — after the last parenthesis — ends the statement, much as a period does in English.

Here's an example of a function call that is written correctly:

say_promise();

But none of these function calls would work:

1st_task(); wouldn't work because it starts with a number

make cake; wouldn't work because in includes a space and no parentheses

do-good (); wouldn't work because it uses a hyphen (or minus sign) which JavaScript doesn't allow, and includes a space between the name and the ()

makeCake()again; wouldn't work because it has text after the parentheses

Arguments make functions more specific.

Let's say that we want to make a chocolate cake. We could write a **makeCake();** function. That would be an instruction to make a cake — but it doesn't say what flavor of cake we want.

We only have to add one argument to specify which type of cake we want. We could code it this way:

makeCake("chocolate");

The makeCake function reads the argument and creates a chocolate cake.

Sometimes a function needs several arguments to work.

For example, let's look at this function:

drawEye();

To draw a green eye that appears in a certain position on a computer screen, we would need to add three arguments:

1. a color
2. an x position
3. a y position.

So we would add those three arguments to the **drawEye();** function and it would look like this:

drawEye("green", 5, 10);

This would draw a green eye, and place it at the (5, 10) coordinates.