**Passing Values to Functions with Arguments**

*Parameters* are variables that act as placeholders for the values that are to be input to a function when it is called. When a function is defined, it is typically defined along with one or more parameters. The actual values that are input (or *"passed"*) into a function when it is called are known as *arguments*.

Here is a function with two parameters, param1 and param2:

function testFun(param1, param2) {

console.log(param1, param2);

}

Then we can call testFun like this: testFun("Hello", "World");. We have passed two string arguments, Hello and World. Inside the function, param1 will equal the string Hello and param2 will equal the string World. Note that you could call testFun again with different arguments and the parameters would take on the value of the new arguments.

1. Create a function called functionWithArgs that accepts two arguments and outputs their sum to the dev console.
2. Call the function with two numbers as arguments.

**Return a Value from a Function with Return**

We can pass values into a function with *arguments*. You can use a return statement to send a value back out of a function.

**Example**

function plusThree(num) {

return num + 3;

}

const answer = plusThree(5);

answer has the value 8.

plusThree takes an *argument* for num and returns a value equal to num + 3.

Create a function timesFive that accepts one argument, multiplies it by 5, and returns the new value.

Run the Tests (Ctrl + Enter)Reset this lesson

Get Help

**Tests**

* Waiting:timesFive should be a function
* Waiting:timesFive(5) should return 25
* Waiting:timesFive(2) should return 10
* Waiting:timesFive(0) should return 0

**Global Scope and Functions**

In JavaScript, *scope* refers to the visibility of variables. Variables which are defined outside of a function block have *Global* scope. This means, they can be seen everywhere in your JavaScript code.

Variables which are declared without the let or const keywords are automatically created in the global scope. This can create unintended consequences elsewhere in your code or when running a function again. You should always declare your variables with let or const.

Using let or const, declare a global variable named myGlobal outside of any function. Initialize it with a value of 10.

Inside function fun1, assign 5 to oopsGlobal ***without*** using the var, let or const keywords.

Run the Tests (Ctrl + Enter)Reset this lesson

Get Help

**Tests**

* Waiting:myGlobal should be defined
* Waiting:myGlobal should have a value of 10
* Waiting:myGlobal should be declared using the let or const keywords
* Waiting:oopsGlobal should be a global variable and have a value of 5