

JS Functions

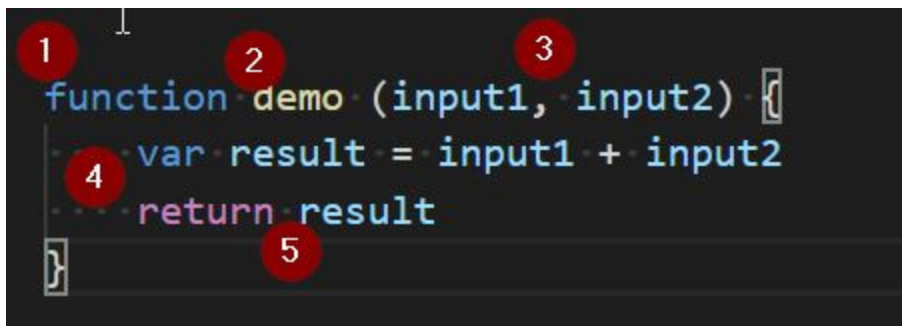
Lesson Time: 30 Minutes

Don't Repeat Yourself

Functions are a block of reusable code. One of the first rules of programming is “Don’t repeat yourself.” If you find yourself re-writing the same lines of code over and over again, it means there is a better way--you can **refactor**, or re-write, the code into a single function and reuse it anytime.

We declare a function with the keyword **function**. Functions can take a list of input parameters and output a result. The inputs we give to a function are called **arguments**. The list of arguments function accepts is called it’s **signature**. When we use a function, we say we are **calling a function**. When we call a function, we must use it’s signature. For example, if a function is defined to take 3 arguments, javascript will throw an error if we only pass in two arguments

A function can return a result with the keyword **return**. A function can also return nothing by omitting the return keyword.



```
1 function demo (input1, input2) {  
2     var result = input1 + input2  
3     return result  
4 }  
5
```

1. The function keyword
2. The name we chose for the function
3. The arguments the function takes as input. This is known as it’s signature.
4. The code block that contains the function’ details
- 5, The return keyword that returns the results

Something that can confuse a new student is the name of the arguments used when creating a function. It's easy to confuse arguments with variables. Remember that the input a function takes is called arguments. Take a look at this example.

```
JS function.js x JS function2.js ... JS function2.js x
23 var y = 3
24 var x = 2
25
26 function thisFunction(x,y){
27   ... var result = x+y
28   ... return result
29 }
30
31 thisVar = thisFunction(10,10)
32 console.log(thisVar)
33
34 thatVar = thisFunction(x,y)
35 console.log(thatVar)]

1 var y = 3
2 var x = 2
3
4 function thisFunction(jelly,peanutButter){
5   ... var result = jelly+peanutButter
6   ... return result
7 }
8
9 thisVar = thisFunction(10,10)
10 console.log(thisVar)
11
12 thatVar = thisFunction(x,y)
13 console.log(thatVar)
```

In the code above on the left, we create variables x and y and assign their values. Next, we define a function, and at first glance, it looks like we are passing x and y. But because we are **defining** the function, this is not the same x and y variables--they are arguments. We can name the arguments anything we want. The code on the right produces the exact same results as the code on the left.

To prove how it works, we can pass 10,10 to our function and get the result 20. Then pass x and y, the variables created at the top of the code, and get 5.

Download the functions.js file attached to this lecture to see an example of functions in action.

Key Terms	Function, arguments, signature, results
Lesson Files	functions.js
Additional Resources	https://www.w3schools.com/js/js_functions.asp

Further Learning	