# INTRODUCTION TO FUNCTIONS

# IN THIS LESSON

What are functions?

Defining a function

Calling a function

Function parameters and arguments

Function return values

Naming function and variables

# WHAT ARE FUNCTIONS?

Another fundamental building  block of JavaScript

A way to write reuseable code

Write code in one place, use it elsewhere as many times as you like

Built-in functions ex. **alert()** and functions we make ourselves

# DEFINING A FUNCTION

One way to define (create) a function is with a function declaration:

```
function sayHello() {
    ...some code...
}
```

The parentheses after the function name contains the parameter list

The parameter list is used to pass values into the function, if needed

In this case, no outside values are needed, so parameter list is empty

# CALLING A FUNCTION

Defining a function does not call (invoke/run/execute) function

One way to call a function from HTML is to use the onclick event handler

To call a function in JavaScript, use function name followed by parentheses

Function Declaration

```
function sayHello() {
    ... code to run ...
}
```

Function Call

```
sayHello();
```

The parentheses in a function call contain the argument list, which parallels the function declaration/definition's parameter list

# FUNCTION PARAMETERS & ARGUMENTS

Function parameters and arguments are how you pass data into a function

Function Declaration w/ parameter of **word**

```
function sayWord(word) {
  alert(word);
}
```

When this function is called, it will automatically declare local variables based on its parameter list - in this case, one local variable named **word**

This variable does not need to be declared with let or const, it only exists inside the function, and it will be automatically deleted when the function ends

Its name doesn't need to be unique within the document, only within the function

Local variables only exist within code block; can't access from outside it; can also be created using let/const

# FUNCTION PARAMETERS & ARGUMENTS

How does a variable declared using a parameter get its value?

Function Declaration w/ parameter of **word**

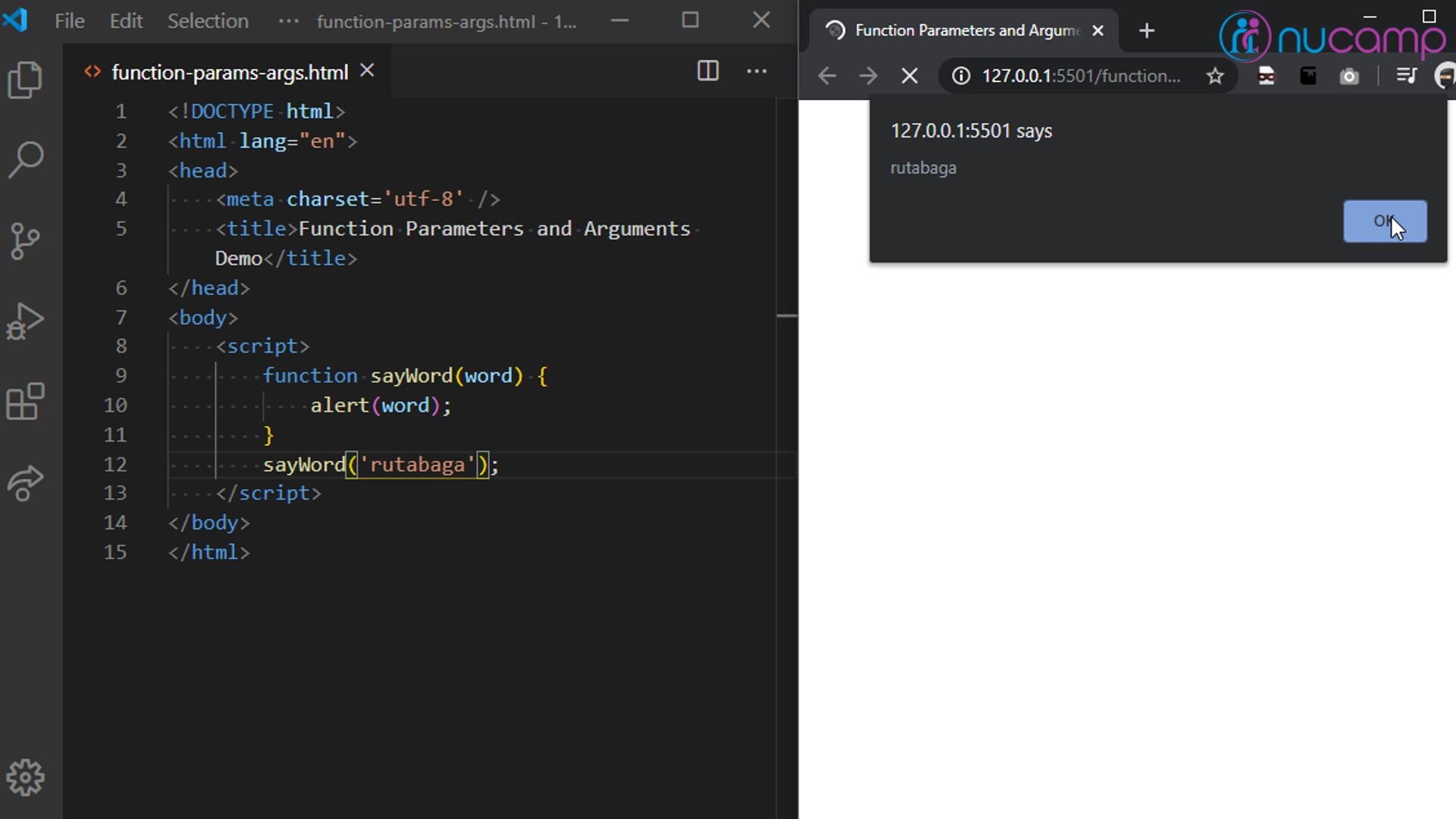
```
function sayWord(word) {
    alert(word);
}
```

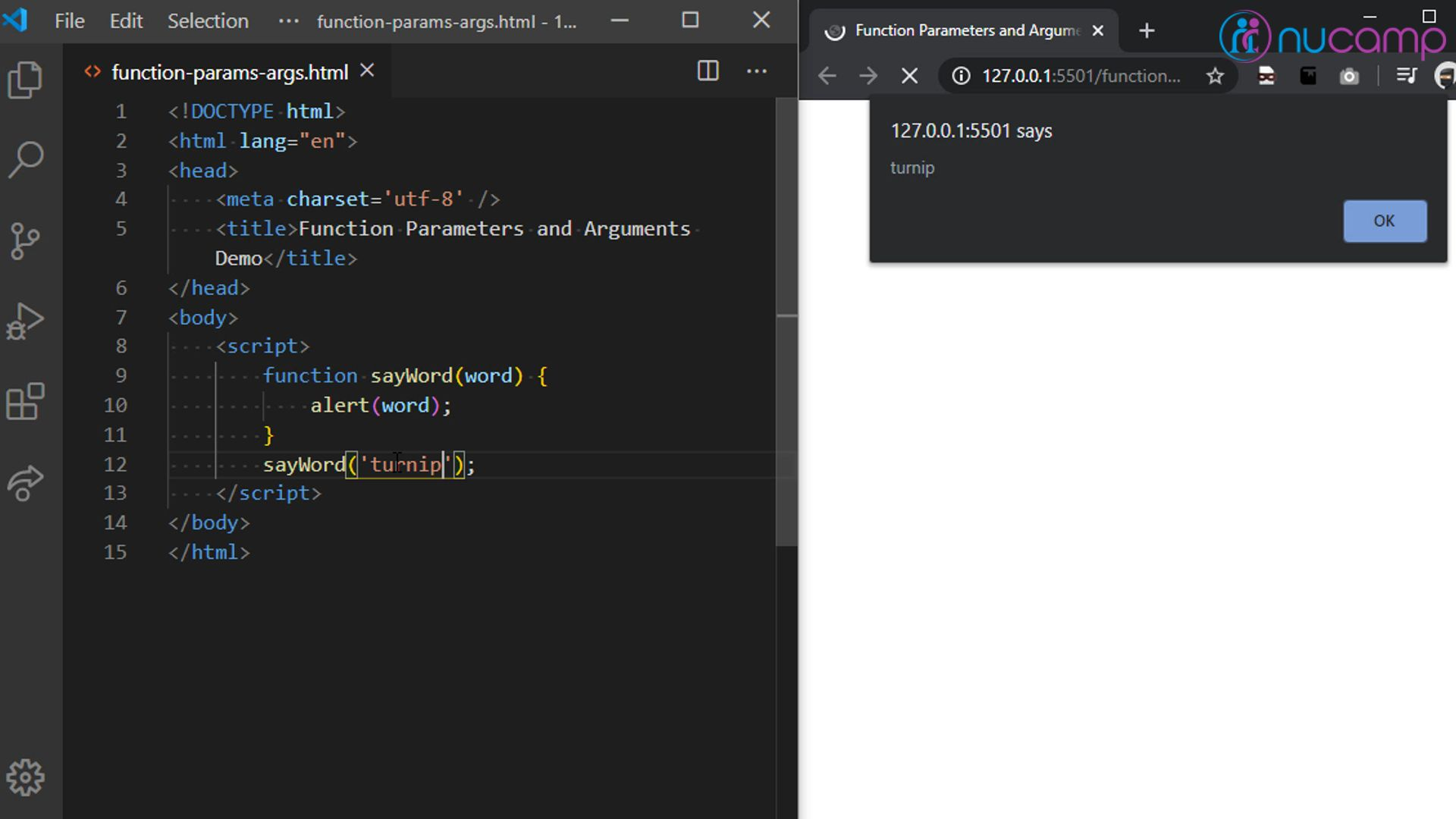
Function Call w/ argument of '**rutabaga**'

```
sayWord('rutabaga');
```

When a function is called, the variables declared inside the function from the parameter list receive their values from the function call's argument list

The assignment is based on order of appearance in the list - '**rutabaga**' will be assigned as the value to the variable '**word**' because they are both first in their respective lists


File Edit Selection
function-params-args.html - 1...
function-params-args.html
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4 <meta charset='utf-8' />
5 <title>Function Parameters and Arguments Demo</title>
6 </head>
7 <body>
8 <script>
9 function sayWord(word) {
10 alert(word);
11 }
12 sayWord('rutabaga');
13 </script>
14 </body>
15 </html>
Function Parameters and Argume
127.0.0.1:5501/function...
127.0.0.1:5501 says
rutabaga
OK
nucamp


File Edit Selection ··· function-params-args.html - 1...
function-params-args.html
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset='utf-8' />
<title>Function Parameters and Arguments Demo</title>
</head>
<body>
<script>
function sayWord(word) {
alert(word);
}
sayWord('turnip');
</script>
</body>
</html>
Function Parameters and Argume
127.0.0.1:5501/function...
127.0.0.1:5501 says
turnip
OK
nucamp

# FUNCTION PARAMETERS & ARGUMENTS

Function Declaration

```
function getArea(width, length) {
    alert(width * length);
}
```

Function Call

```
getArea(3, 4);
```

Function parameters  define the names of variables declared locally inside the function, whose values are provided from outside of the function

Function arguments are used in function calls to provide those values

# FUNCTION RETURN VALUES

Functions always return a single value; if you do not provide that value, it returns the value of **undefined**

To return a value, use the **return** keyword followed by the value

The value is returned to where the function was called

You can call the function and assign its return value to a variable:

```
function getArea(width, length) {
    return width * length;
}
```

```
let area = getArea(3, 4);
alert(area);
```

# NAMING FUNCTIONS AND VARIABLES

Naming convention for JavaScript functions and variables: camelCase

Use lower case first letter, upper case for any additional words:

```
const studentFirstName = "Ronald";
let currentHighScore = 12345;
function getLevelNames() { ... }
```

Some exceptions - ex.: for variables not meant to be set or changed during a program's execution, use SCREAMING_SNAKE_CASE

```
const MAX_ITEMS = 10;
```

Other languages may use different conventions, e.g. snake_case, kebab-case

Be aware that JavaScript is case sensitive