Upright Education

JavaScript Functions

Software Development Bootcamp

Understanding and Using Functions in JavaScript

Upright Education

Topic

Functions

What are Functions?

Functions are reusable blocks of code that perform a specific task.

Functions can:

- Accept input (parameters)
- Process data
- Return output (results)

Think of functions as “action” in your code, for example

- `calculateTotal()`
- `sendEmail()`
- `validateUserInput()`

Why Use Functions?

- Organization & Reusability
  - Break complex problems into smaller, manageable pieces
  - Write code once, use it many times
- Abstraction
  - Hide complex operations behind simple interfaces
- Modularity
  - Develop and test parts of your program independently
  - Easier debugging and collaboration
- Scalability
  - Easily extend and modify your code as your project grows

The `return` Keyword

The **return** statement ends function execution and specifies the value to be returned by the function.

- Optional: functions without `return` implicitly return `undefined`
- Once return is executed the function immediately stops.
- You can have multiple `return` statements in a function (but only one will be executed)

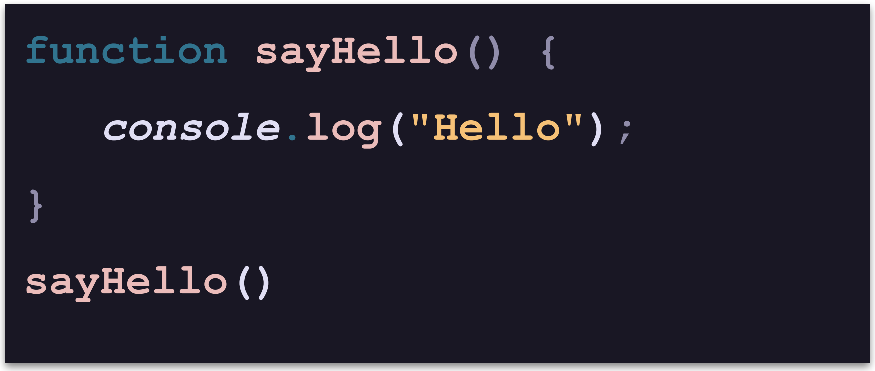
Function Syntax

- `function`: The function keyword tells JavaScript you're creating a function
- `functionName`: This is the name of the function. You can name it almost anything you like.
- `Parameters`: These are inputs to the function, you can have multiple parameters separated by commas. If there are no inputs, you can leave this empty.
- `{...}`: Curly braces contain the code that runs when the function is called.

```
function functionName(parameters)
{
    // Code to be executed
}
```

Simple Function Example

- `sayHello`: This is the functions name
- `()`: No parameters are needed for this function
- `console.log(“hello”)`: This is the code that will run when we call the function.
- `sayHello()`: To use the function, you call it by its name followed by parentheses



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

Upright Education

Function with Parameters

- **`name`**: This is a parameter. It is a placeholder for the actual input you'll give the function.
- **`greet('Jane')`**: Calling the `greet` function with the argument `'Jane'`

```
function greet(name) {
   console.log("Hello, " + name +
"!");
}
greet('Jane')
```

Passing Variables to Functions

When passing a variable to a function, its value is assigned to a parameter. Variable and parameter names do not need to match.

```
function shout(someString) {
   let loudString = someString.toUpperCase();
   return loudString + '!!!'
}
let myString = 'hello world'
let myShoutString = shout(myString)
console.log(myString)
console.log(myShoutString)
```

shout Function Explained

- **`function shout(someString)`**: Defines a function named `shout` that takes one parameter, `someString`
- **`let loudString = someString.toUpperCase()`**: Inside the function we create a variable `loudString`. The `toUpperCase()` method is called, converting all the characters in the string to uppercase letters.
- **`return loudString + !!!`**: Returns the value of `loudString` with three exclamation marks added to the end.

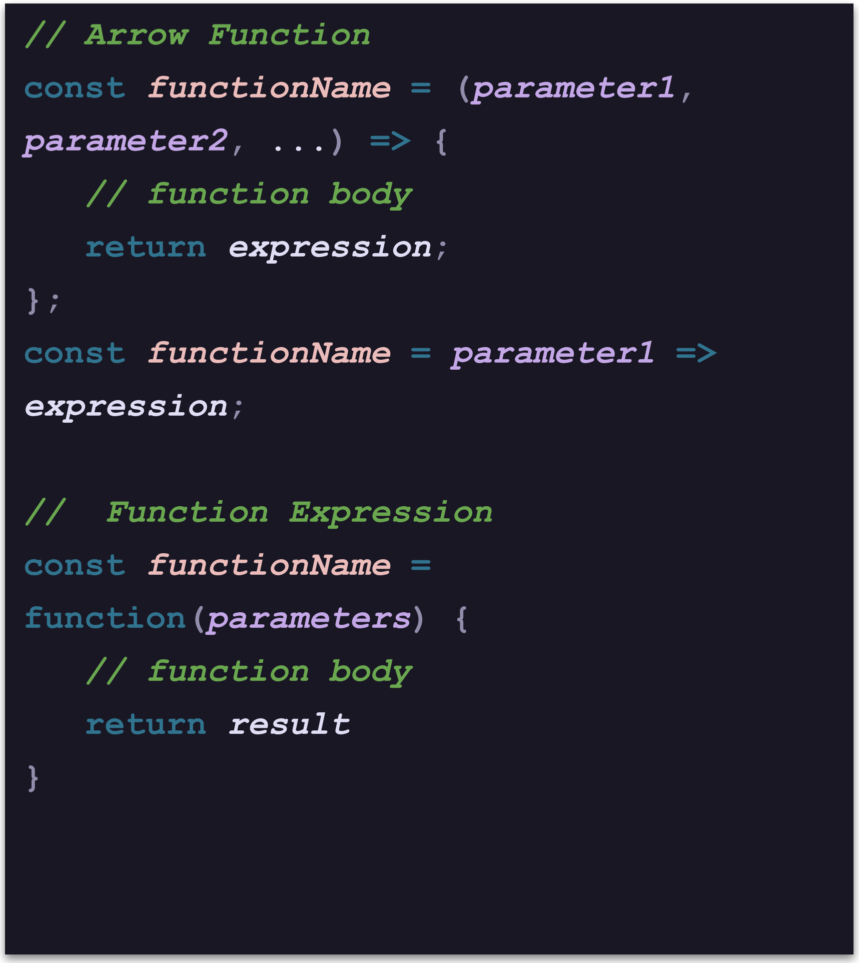
Using the shout Function

- **`let myString = ‘hello world``**: Creates a variable `myString` and assigns it the value “hello world”
- **`let myShoutString = shout(myString)`**: Calls the `shout` function with `myString` as the argument. The function converts “hello world” to “HELLO WORLD” and adds “!!!”, so the value of `myShoutString` will be “HELLO WORLD!!!”

Alternative Ways to Write Functions

- Arrow Functions: Provide a concise way to write functions in JavaScript.
  - If there is only one parameter and the function body only contains a single expression, you can omit the parentheses and the curly braces

- Function Expression: A function expression defines a function as part of an expression.
  - Cannot be called before it is defined.



```
// Arrow Function
const functionName = (parameter1,
parameter2, ...) => {
   // function body
   return expression;
};
const functionName = parameter1 =>
expression;

//  Function Expression
const functionName =
function(parameters) {
   // function body
   return result
}
```

Upright Education

Exercise

Hello Frenemy