

MODULE 4

JavaScript Functions

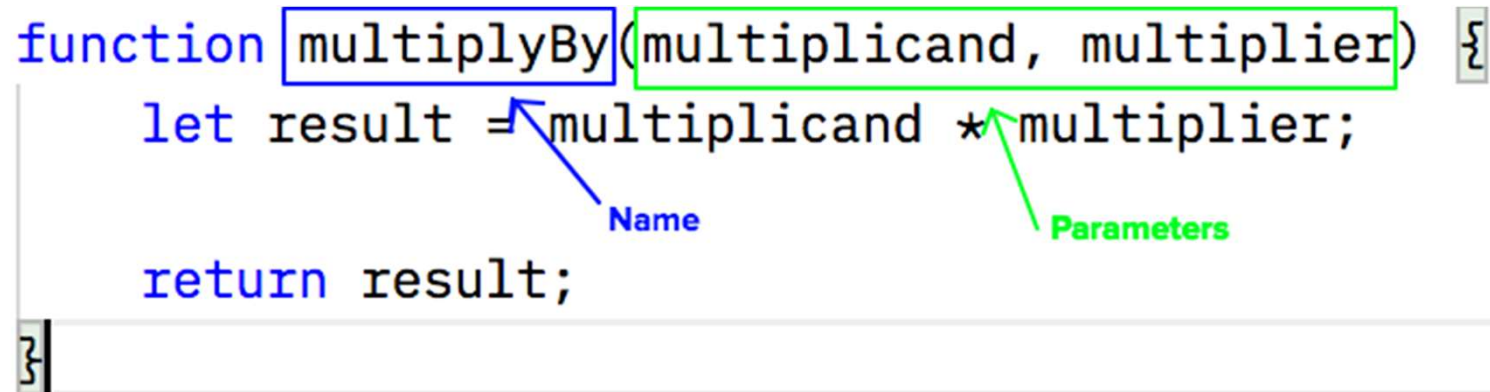


What is a function?



Javascript functions

```
function multiplyBy(multiplicand, multiplier) {  
    let result = multiplicand * multiplier;  
    return result;  
}
```



- **descriptive** - it should be clear what type of action or calculation the function performs when invoked
- **camelCase** - the first letter of the name is lowercase and the first letter of each subsequent word is uppercase
- **unique** - function names need to be unique across all JavaScript code that is loaded into the page. If a name conflicts with another function, the one that's loaded last will overwrite the other one

Function Parameters

```
function multiplyBy(multiplicand, multiplier) {  
  let result = multiplicand * multiplier;  
  
  return result;  
}
```



- Parameters are optional!
- Can set defaults
- Where are the parameter data types?

We don't need no stinkin' parameters

- You can give a function as many parameters as you want.
- If no parameters are defined, you can still pass parameters.
- Use arguments array to access them.

```
function concatAll() { // No parameters defined, but we still might get some
  let result = "";
  for(let i = 0; i < arguments.length; i++) {
    result += arguments[i];
  }
  return result;
}
```

Anonymous Functions

Parameters  **Fat Arrow** 

```
(multiplicand, multiplier) => {  
  let result = multiplicand * multiplier;  
  
  return result;  
}
```

Anonymous Functions

```
let multiply = (multiplicand, multiplier) => {  
  let result = multiplicand * multiplier;  
  
  return result;  
}  
  
console.log( multiply(5, 2) ); // Prints `10` to the console
```

Anonymous Functions

```
// Filter an array of numbers so that we are only left with even numbers
let numbers = [1, 2, 3, 4];

let evenNumbers = numbers.filter( (number) => {
    return number % 2 === 0;
});

console.log( evenNumbers ); // Prints out `[2, 4]`
```


Anonymous Functions

```
// Filter an array of numbers so that we are only left with even numbers
```

```
let numbers = [1, 2, 3, 4];
```

What we want to do

```
let evenNumbers = numbers.filter( (number) => {
```

```
  return number % 2 === 0;
```

```
});
```

What rules to apply

```
console.log( evenNumbers ); // Prints out `[2, 4]`
```

What to operate on

filter

```
let numbersToFilter = [1, 2, 3, 4, 5, 6];

let filteredNumbers = numbersToFilter.filter( (number) => {
  // Only keep numbers divisible by 3
  return number % 3 === 0;
});

console.log(filteredNumbers);
```

forEach

```
let numbers = [1, 2, 3, 4];
```

```
numbers.forEach( (number) => {  
  console.log(`This number is ${number}`);  
});
```

map

```
let numbersToSquare = [1, 2, 3, 4];
```

```
let squaredNumbers = numbersToSquare.map( (number) => {  
    return number * number;  
});
```

```
console.log(squaredNumbers);
```

reduce

```
let nameParts = ['bosco', 'p.', 'soultrain'];
```

```
let fullName = nameParts.reduce( (reducer, part) => {  
    return reducer + ' ' + part.substring(0, 1).toLocaleUpperCase() +  
    part.substring(1);  
}, ""); // <--- The empty quotes is the value of the reducer for the first  
element
```

```
console.log(fullName.trim());
```

A visual to remember...

```
map([🐮, 🍌, 🐔, 🌽], cook)  
=> [🍔, 🍟, 🍗, 🍿]
```

```
filter([🍔, 🍟, 🍗, 🍿], isVegetarian)  
=> [🍟, 🍿]
```

```
reduce([🍔, 🍟, 🍗, 🍿], eat)  
=> 💩
```

Passing a Function as Parameter

```
function addContact(id, refreshCallback) {  
    refreshCallback();  
}
```

```
function refreshContactList() {  
    alert('Hello World');  
}
```

```
addContact(1, refreshContactList);
```

LET'S CODE!



ELEVATE  YOURSELF

WHAT QUESTIONS DO
YOU HAVE?



Reading for tonight:

The DOM

