



Codergirl - JavaScript

Class 7

September 9, 2020

Agenda

- Studio recap – 6:10 pm
- Lecture – 6:40 pm
- Exercise – 7pm



Studio recap

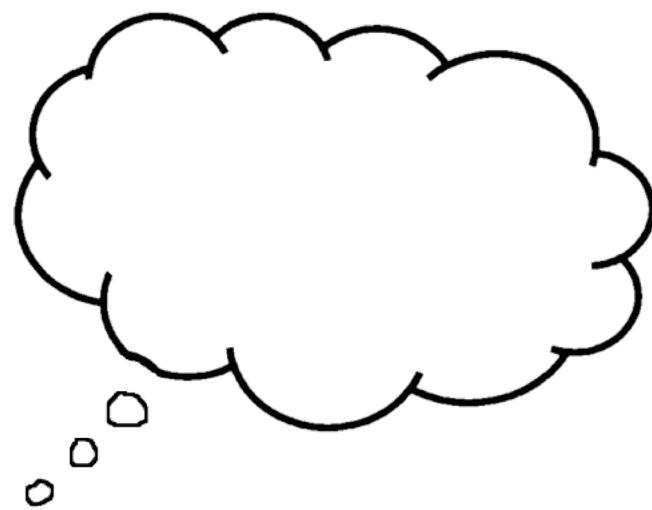
Sort an Array

1. Find the minimum value in an array
2. Loop through the array, call the function that finds the min value.
3. Remove it from original array and add to the new array.
4. Repeat loop until original array is empty.

OR

- Use the Array sort function!
- Bonus: Sort using recursion.

...



...

Array spread operator

```
let numbers1 = [1, 2, 3, 4, 5];
```

```
let numbers2 = [ ...numbers1, 1, 2, 6, 7, 8];
```

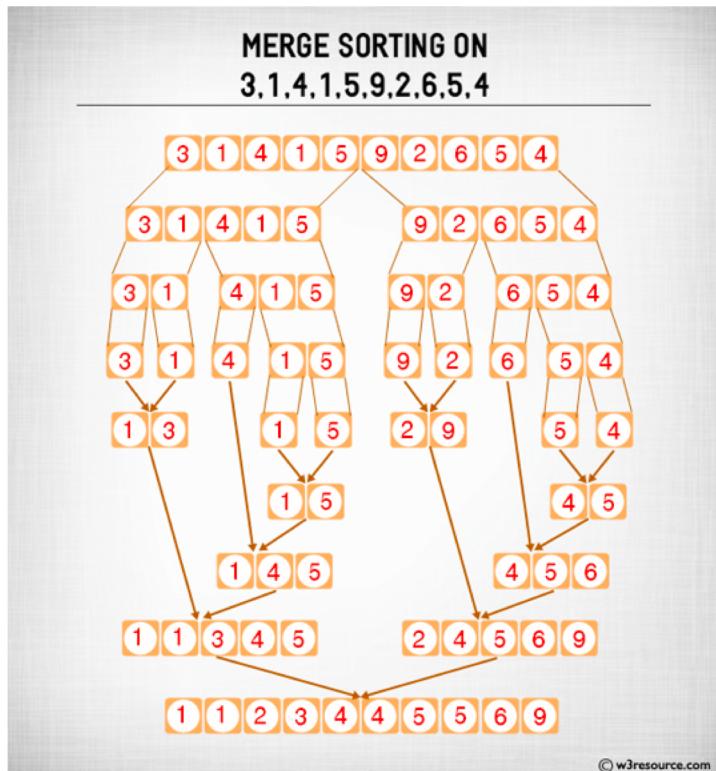
```
// this will be [1, 2, 3, 4, 5, 1, 2, 6, 7, 8]
```

Array Shift

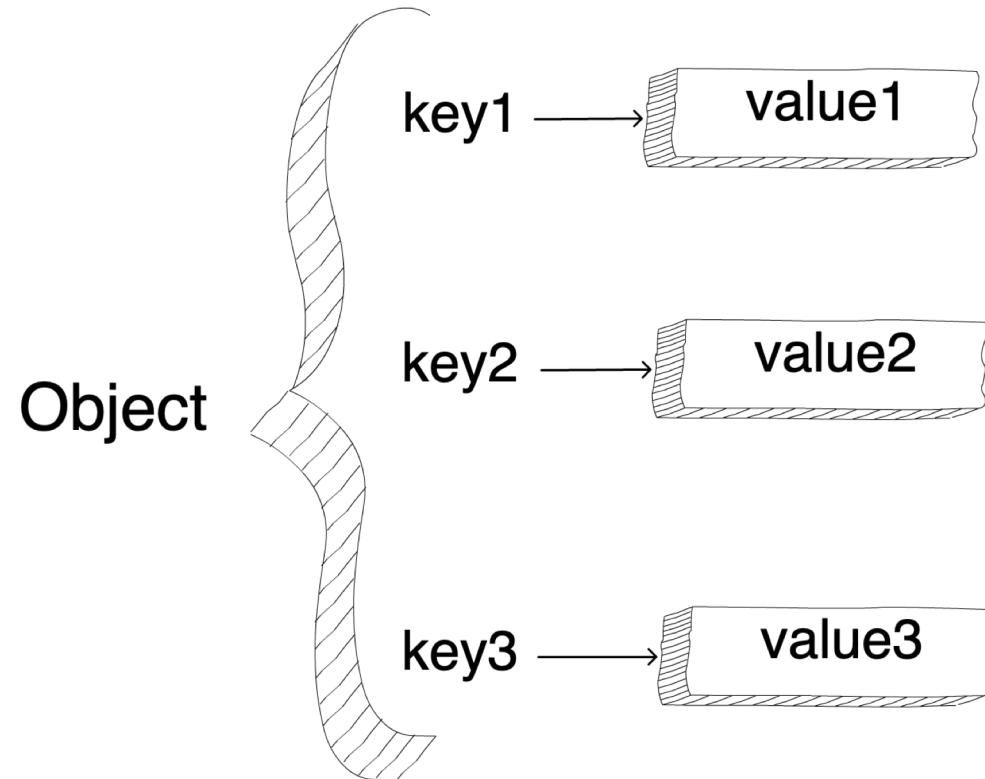
```
let array1 = [1, 2, 3];  
let firstElement = array1.shift();  
  
console.log(array1);  
// expected output: Array [2, 3]  
  
console.log(firstElement);  
//expected output: 1
```

Whiteboard

Recursion sorting



Objects



Example of Object

```
example.js > person > signature

let person = {
    name: "Tintin",
    age: 22,
    id: 1234,
    country: "Belgium",
    email: "tintinthereporter@gmail.com",
    hobbies: ["solving mysteries", "playing with Snowy", "meeting Captain Haddock"],
    signature: function() {
        return `id: ${id} name: ${name}`;
    }
}
```

Working with Objects

this keyword can be used to refer to an object within an object
shorthand for objects name.

Accessing properties

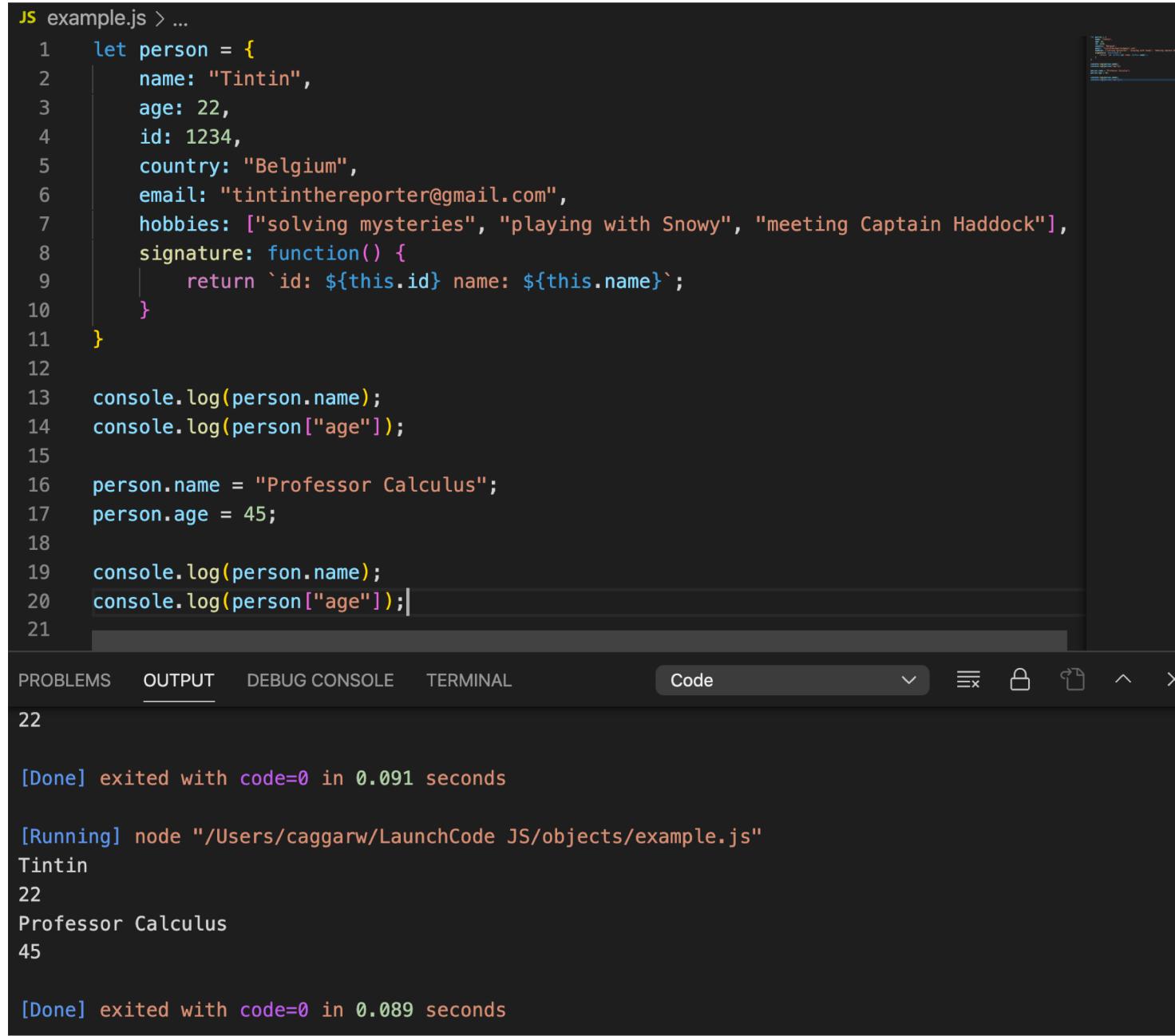
```
JS example.js > [⌚] person
1  let person = [
2    name: "Tintin",
3    age: 22,
4    id: 1234,
5    country: "Belgium",
6    email: "tintinthereporter@gmail.com",
7    hobbies: ["solving mysteries", "playing with Snowy", "meeting Captain Haddock"],
8    signature: function() {
9      return `id: ${this.id} name: ${this.name}`;
10 }
11 }
12
13 console.log(person.name);
14 console.log(person["age"]);
```

The screenshot shows a dark-themed interface of the Visual Studio Code code editor. At the top, there's a status bar with tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL, followed by a dropdown labeled 'Code'. Below the editor area, the terminal window displays the output of running the script:

```
[Running] node "/Users/caggarw/LaunchCode JS/objects/example.js"
Tintin
22

[Done] exited with code=0 in 0.091 seconds
```

Modifying properties



The screenshot shows a dark-themed code editor interface with a sidebar on the right containing file navigation and status information. The main area displays a JavaScript file named 'example.js'.

```
JS example.js > ...
1 let person = {
2   name: "Tintin",
3   age: 22,
4   id: 1234,
5   country: "Belgium",
6   email: "tintinthereporter@gmail.com",
7   hobbies: ["solving mysteries", "playing with Snowy", "meeting Captain Haddock"],
8   signature: function() {
9     return `id: ${this.id} name: ${this.name}`;
10 }
11 }
12
13 console.log(person.name);
14 console.log(person["age"]);
15
16 person.name = "Professor Calculus";
17 person.age = 45;
18
19 console.log(person.name);
20 console.log(person["age"]);|
21
```

The code defines an object 'person' with properties like name, age, id, country, email, hobbies, and a signature function. It then logs the original values of name and age to the console. Afterward, it changes the values of name and age to new ones ('Professor Calculus' and 45 respectively) and logs them again to the console.

Below the code editor, a terminal window shows the execution results:

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL Code ▾ 🔍 ⌂ ⌄ ⌁ ×
22
[Done] exited with code=0 in 0.091 seconds
[Running] node "/Users/caggarw/LaunchCode JS/objects/example.js"
Tintin
22
Professor Calculus
45
[Done] exited with code=0 in 0.089 seconds
```

Adding properties

```
JS example.js > ...
1  let person = {
2      name: "Tintin",
3      age: 22,
4      id: 1234,
5      country: "Belgium",
6      email: "tintinthereporter@gmail.com",
7      hobbies: ["solving mysteries", "playing with Snowy", "meeting Captain Haddock"],
8      signature: function() {
9          return `id: ${this.id} name: ${this.name}`;
10     }
11 }
12
13 person.sex = 'M';
14
15 console.log(person);
16

PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL
[Running] node "/Users/caggarw/LaunchCode JS/objects/example.js"
{ name: 'Tintin',
  age: 22,
  id: 1234,
  country: 'Belgium',
  email: 'tintinthereporter@gmail.com',
  hobbies:
   [ 'solving mysteries',
     'playing with Snowy',
     'meeting Captain Haddock' ],
  signature: [Function: signature],
  sex: 'M' }
```

Coding with Objects

```
JS exampleComparison.js > ...
1  let detective1 = {
2    name: "Tintin",
3    age: 22
4  }
5
6  let detective2 = {
7    name: "Tintin",
8    age: 22
9  }
10
11 console.log(`detective1 === detective2 is ${detective1 === detective2}`);
12 console.log(`detective1 == detective2 is ${detective1 == detective2}`);
13 console.log(`detective1.age === detective2.age is ${detective1.age === detective2.age}`);
14 console.log(`detective1.name === detective2.name is ${detective1.name === detective2.name}`);
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

[Running] node "/Users/caggarw/LaunchCode JS/objects/exampleComparison.js"

```
detective1 === detective2 is false
detective1 == detective2 is false
detective1.age === detective2.age is true
detective1.name === detective2.name is true

[Done] exited with code=0 in 0.095 seconds
```

Code

loop

```
1 let person = {  
2   name: "Tintin",  
3   age: 22,  
4   id: 1234,  
5   country: "Belgium",  
6   email: "tintinthereporter@gmail.com",  
7   hobbies: ["solving mysteries", "playing with Snowy", "meeting Captain Haddock"]  
8 }  
9  
10 for (item in person) [  
11   console.log(item + ': ' + person[item]);  
12 ]
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL **Code**
[Running] node "/Users/caggarw/LaunchCode JS/objects/exampleFor.js"
name: Tintin
age: 22
id: 1234
country: Belgium
email: tintinthereporter@gmail.com
hobbies: solving mysteries, playing with Snowy, meeting Captain Haddock

[Done] exited with code=0 in 0.089 seconds

• • •

Object spread operator

```
JS exampleObjectSpread.js > ...
1  let person = {
2    name: "Tintin",
3    age: 22,
4    id: 1234,
5    country: "Belgium",
6  }
7
8  let tintin = {
9    ...person,
10   sex: "M"
11 }
12
13 console.log(person);
14 console.log(tintin);
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

Code

```
[Running] node "/Users/caggarw/LaunchCode JS/objects/exampleObjectSpread.js"
{ name: 'Tintin', age: 22, id: 1234, country: 'Belgium' }
{ name: 'Tintin', age: 22, id: 1234, country: 'Belgium', sex: 'M' }
```

Math

built in object – 8 properties
methods

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Math

Not modifiable

Math methods

[2. MDN Web Docs ↗](#)

To see detailed examples for a particular method, click on its name.

Ten Common Math Methods

Method	Syntax	Description
abs	<code>Math.abs(number)</code>	Returns the positive value of <code>number</code> .
ceil	<code>Math.ceil(number)</code>	Rounds the decimal <code>number</code> UP to the closest integer value.
floor	<code>Math.floor(number)</code>	Rounds the decimal <code>number</code> DOWN to the closest integer value.
max	<code>Math.max(x,y,z,...)</code>	Returns the largest value from a set of numbers.
min	<code>Math.min(x,y,z,...)</code>	Returns the smallest value from a set of numbers.
pow	<code>Math.pow(x,y)</code>	Returns the value of x raised to the power of y (x^y).
random	<code>Math.random()</code>	Returns a random decimal value between 0 and 1, NOT including 1.
round	<code>Math.round(number)</code>	Returns <code>number</code> rounded to the nearest integer value.
sqrt	<code>Math.sqrt(number)</code>	Returns the square root of <code>number</code> .
trunc	<code>Math.trunc(number)</code>	Removes any decimals and returns the integer part of <code>number</code> .

Exercise

Questions?

<https://www.amazon.com/How-Women-Rise-Holding-Promotion/dp/0316440124>

PART II
**The Habits That Keep Women from
Reaching Their Goals**

4. <u>The Twelve Habits</u>	<u>47</u>
5. <u>Habit 1: Reluctance to Claim Your Achievements</u>	<u>63</u>
6. <u>Habit 2: Expecting Others to Spontaneously Notice and Reward Your Contributions</u>	<u>76</u>
<hr/>	
7. <u>Habit 3: Overvaluing Expertise</u>	<u>86</u>

Studio time!