JavaScript Objects

Learning Objectives

- Understand what objects are
- Understand how to create and use objects

Material

Objects

Objects are a feature of most modern programming languages. They allow us to encapsulate related variables and functions together. Objects can be created in JavaScript using curly braces:

```
let myObj = {};
```

The object above is empty. We can add properties to it using the dot or square bracket notation:

```
let myObj = {};

myObj["a"] = 1;
myObj.b = 2;

console.log(myObj);
// { a: 1, b: 2 }
```

We can also just create objects with properties directly:

```
let person1 = {
    name: "Meredith",
    age: 14
}
```

We can access the values in an object with the same syntax we use for assigning:

```
let person1 = {
```

```
    name: "Meredith",
    age: 14,
};
console.log(person1.age);
console.log(person1["name"]);
// 14
// Meredith
```

Objects can also have functions as properties. These are called *methods*.

```
let tiger = {
    age: 3,
    greet: function () {
        console.log("roar");
    },
};
tiger.greet();
// roar
```

Methods on objects can refer to the parent object using the `this` keyword:

```
let person1 = {
    name: "Mandy",
    greet: function () {
        console.log("Hello, I am", this.name);
    },
};
person1.greet();
// Hello, I am Mandy
```

Objects can also refer to other objects:

```
let user1 = {
    name: "Jane",
    age: 24,
};

let comment = {
    text: "omg that looks amazing",
    date: "06/08/2021",
    author: user1,
};
console.log(comment.author.name);
// Jane
```

Core Assignments

Family Tree

Create objects to represent 3 generations of a family. You can pick any family you like (Starks, Simpsons, Royal Family etc.). A person in the family tree should have the following properties:

- `name` - the person's name as a string
- `parents` - a 2-element array containing references to the person's parent objects
- `printParents()` - a function that prints the names of the person's parents

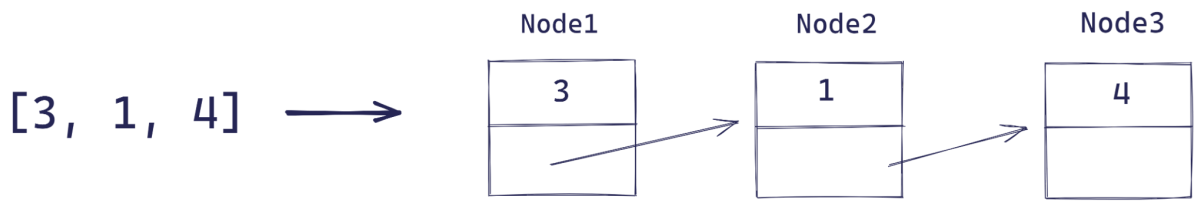
Once you've created your family, check that it's working e.g.

```
// your code above

console.log(sansa.name)
// Sansa

console.log(sansa.parents)
// [Object, Object]

sansa.printParents()
// Ned
// Catelyn
```

Extension Assignments

Linked List

A fundamental data structure in computer science is a linked list. A linked list is made of nodes. Each node in the list is an object and contains a value as well as a reference to the next node in the list. Write a function that takes an array of numbers as an argument and creates a linked list of the same numbers. The function should return the first node in the list.



Once you've created a linked list, see if you can iterate over it.

Linked List++

Create a function which, given a linked list, partitions the list around a particular value, x. Nodes with values less than x should appear at the start of the list. Nodes with values greater than or equal to x should appear at the end of the list.

```
e.g. x = 7

1->4->19->3->8->16->4   --->   1->4->3->4->19->8->16
```

Additional Resources

- W3schools object guide