

JavaScript Classes and Object Creation

In this lesson, we briefly look at object-oriented programming concepts in JavaScript like classes and objects.

Background

Classes are relatively new to JavaScript as previously, there was only JavaScript's [prototype chain](#) which could be used for inheritance too. JavaScript classes build up on top of the prototypical inheritance giving the whole thing a more straightforward representation. Classes in JavaScript are syntactical sugar over prototype chains. You don't need to understand what a prototype chain or prototypical inheritance is for this course but knowing these concepts would give you some context of classes in the language.

What is a class?

A class is a kind of separate mini-program with a context of its own — methods (functions) and properties (variables). They describe an entity in a way that a computer can understand. Disparate 'instances' (called objects) of a class can be created and treated as variables. Let's examine the following code to learn how classes work in JavaScript.

```
class Developer {
  constructor(firstname, lastname) {
    this.firstname = firstname;
    this.lastname = lastname;
  }

  getName() {
    return `${this.firstname} ${this.lastname}`;
  }
}

var me = new Developer('Robin', 'Wieruch');

console.log(me.getName());
```



Class Object Creation & Initializing Class Variables

The entity described in this example is a ‘Developer.’ A new object can be created with the `new` keyword and the ‘constructor’ of the class can be called at the same time as on line 12 to assign values to the class’s properties, namely the `firstname` and the `lastname`. The `firstname` of the `me` instance of the class `Developer` is assigned the value ‘Robin,’ and the `lastname` is assigned ‘Wieruch.’






Also, various class methods such as `getName()` can be used to read or write the properties of the object. The `console.log()` method prints whatever is passed as an argument to it.

That’s all you need to understand React class components. A JavaScript class is used for defining a React component, but as you will see in the next lesson, the React component is only a React component because it inherits all the abilities from the `_React Component class_` which is imported from the React package.

Coding Challenge: Create a JavaScript class

In the following coding challenge, write a class in JavaScript that describes a car with the properties `color`, `model`, `engineCap` (engine capacity), and `registrationNum` registration number and methods `getColor()`, `getModel()`, `setColor()`, and `setModel()`. Remember to use the exact spelling given here or your code won’t pass.

// Write-your-code-here



Object Creation in JavaScript

Classes — as mentioned previously — are relatively new to Javascript. Objects are a lot like classes. Have a look at the code sample below. Here, we define three objects `computer`, `computer2` and `computer 3`. Each of them has the properties `brand`, `RAM` and `clockspeed`. In addition, `computer3` has the method `printRam()`. Note that objects can be defined on one line as `computer` is

defined or can span multiple lines as the other two are.

```
let computer = { brand : 'HP', RAM : '8 GB', clockspeed : "2 GHz"};

// object definitions can have spaces and newlines!
let computer2 = {
  brand : 'HP',
  RAM : '8 GB',
  clockspeed : "2 GHz"
};

// Objects can also have 'functions' called methods
let computer3 = {
  brand : 'HP',
  RAM : '8 GB',
  clockspeed : "2 GHz",

  printRam() {
    console.log(this.RAM)
  }
}
```



You may have noticed the keyword `this` used in the class declarations, we'll study it in depth in the next lesson.