Lesson Weekend

Intermediate JavaScript (/intermediate-javascript) / Object-Oriented JavaScript (/intermediate-javascript/object-oriented-javascript) / Literal Notation Versus Constructors

Text | Cheat sheet

In the previous lesson we created objects using **Literal Notation**. Let's explore further examples of what literal notation looks like, and how the process of creating objects may be streamlined by using constructors. We haven't learned about constructors yet, but we will in this lesson!

Literal Notation

Let's say a dog walker wants to keep track of all dogs they walk. We'll create a dog object using literal notation:

```
let dog1 = {
  name: "Falcor",
  colors: ["black"],
  age: 4
};
```

Now we'll create another:

```
let dog2 = {
  name: "Nola",
  colors: ["white", "black"],
  age: 6
};
```

And another:

```
let dog3 = {
  name: "Patsy",
  colors: ["brown"],
  age: 2
};
```

You may notice that this is getting a little repetitive. All of these dogs have the same properties, which involves typing `name`, `colors` and `age` over and over again, each time we want to create a new dog. Good news is, there is a much faster way to make dog objects! Instead of using literal notation to manually create each individual dog object, we can use a constructor as a blueprint.

Constructors

We will write a **constructor function** to create dog objects momentarily, but first, what *is* a constructor function? A constructor function (or just 'constructor') is a special type of function that is used to make an instance of a type of object, like a `Dog` object. The next lesson will introduce constructors and demonstrate how we can use constructors and prototypes to make our lives easier.

After we write a constructor we'll be able to create the same dogs we created in literal notation above, with these three simple lines of code:

```
let dog1 = new Dog("Falcor", ["black"], 4);
let dog2 = new Dog("Nola", ["white", "black"], 6);
let dog3 = new Dog("Patsy", ["brown"], 7);
```

As you can see, this is *far* less code, and it's much more scalable!

Previous (/intermediate-javascript/object-oriented-javascript/javascript-objects)
Next (/intermediate-javascript/object-oriented-javascript/constructors-and-prototypes)

Lesson 5 of 33
Last updated March 23, 2023

disable dark mode

(http://www.epicodus.com)