Get & Set

This lesson teaches us how to use the "get" and "set" keywords in JavaScript.

We'll cover the following

Using `get` Keyword

In the previous lesson we discussed the following code:

```
var employee = {

  name: 'Joe',
  age: 28,
  designation: 'developer',
  //function returning designation of the employee
  display() {
    return this.designation //using this to refer to the "employee" object
  }
}
//this will display the designation
console.log(employee.display())
```

Here, the function `display()` was being used to *get* the value of the *property* `designation`. Another way to do this is by using the `get` keyword.

Example

Let's take a look at an example implementing the `get` *keyword*.

```
 1  var employee = {
 2  
 3    name: 'Joe',
 4    age: 28,
 5    designation: 'developer',
 6    //function returning designation of the employee
 7    get display() {
 8      return this.designation //using this to refer to the "employee" object
 9    }
10  }
11  //this will display the designation
12  console.log(employee.display)
```

Explanation

You must be wondering what the difference is since the function definition is exactly the same as before, except for the use of the *keyword* `get`. Now, look closely at **line 12** in both the code executables above.

See the difference?

Using `get` changes the way the *function* `display()` is called. It is now called in exactly the same way as how a *property* is called: `employee.display`, whereas without `get`, it is called as a *function*: `employee.display()`.

Using `set` Keyword

In the previous lesson, we learned how to use `this` to set the value of a property inside an object. We discussed the following code:

```
var employee = {

  name: 'Joe',
  age: 28,
  designation: 'developer',
  //function setting the value of "designation" equal to the parameter being passed to the fu
  setDesignation(desig) {
    this.designation = desig
  }
}
//displaying the value of "designation" at start
console.log("Old designation was:",employee.designation)
//updating the value of designation
```

```
employee.setDesignation('engineer')
//displaying new value of designation
console.log("New designation is:",employee.designation)
```

Another way to do this is by using the `set` *keyword*.

Example

Let's take a look at an example using the `set` keyword below:

```
var employee = {

  name: 'Joe',
  age: 28,
  designation: 'developer',
  //function setting designation of the employee
  set setDesignation(desig) {
    this.designation = desig //using this to refer to the "employee" object
  }
}
console.log("designation originally is:",employee.designation)
employee.setDesignation = 'engineer'
console.log("new designation is:",employee.designation)
```

Explanation

Using the `set` *keyword* changes the way `setDesignation` is used in order to set the `designation` value.

Previously, the value of `designation` was set by the approach shown in **line 14** of the first code executable, i.e., by calling `setDesignation` as a *function* and passing the parameter `engineer` to it. However, looking at **line 12** of the code widget above shows that when the `set` *keyword* is used, `setDesignation` sets the value of `designation` similarly to how any other property value would be set.

In conclusion, as seen from the above examples, `get` and `set` allow functions to be accessed and changed as data values outside the object.

Now that you’ve learned about *objects* in JavaScript let’s put that knowledge to test in the next lesson!