Accessing Object Properties

This lesson discusses the different ways to access properties, such as using dot operator and square brackets.

WE'LL COVER THE FOLLOWING

There are various ways to access object properties. We will discuss each method in detail below.

Dot Operator

In JavaScript, an object literal can be accessed using the **dot operator**. To access any property, the *name* of the object should be mentioned first, followed by the *dot operator* and then the *name* of the *property* encapsulated in that object.

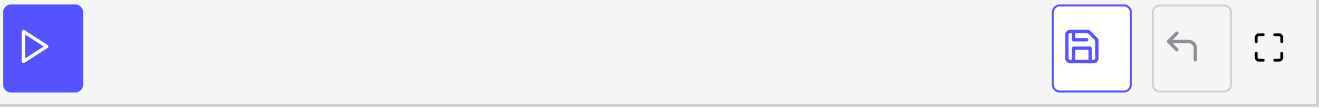
Example

Let's take a look at an example:

```
//creating an object named shape


var shape = {
  //defining properties of the object
  //setting data values
  name : 'square',
  sides : 4
}

//accessing the properties using the dot operator
console.log("Name is:", shape.name) //using dot operator to access "name"
console.log("Number of sides are:", shape.sides) //using dot operator to access "sides"
```



Accessing Properties of a Nested Object

If the value of a property of an object is another object, how will it be accessed? The method is exactly the same with the addition of just one more dot, i.e., object name followed by dot operator followed by the second object name, then the second dot operator and lastly, the name of the property.

Example

Let's take a look at an example for accessing the property of an object nested inside another object.

```
//creating an object named employee

var employee = {
  //defining properties of the object
  //setting data values

  //value of "name" is another object
  name : {
    firstName: 'Joe',
    lastName : 'Adams'
  },
  age : 28,
  designation : 'Developer'
}

//accessing the first name
console.log("First name is:", employee.name.firstName)

//accessing the last name
console.log("Last name is:", employee.name.lastName)

//accessing the age
console.log("Age is:", employee.age)

//accessing the designation
```

```
//accessing the designation
console.log("Designation is:",employee.designation)
```

Using Square Brackets

Another method to access values is by using square brackets `[]`. The name of the *property* to be accessed is written as a string inside the square brackets.

Example

Below is an example using square brackets to access a property.

```
//creating an object named shape

var shape = {
  //defining properties of the object
  //setting data values
  name : 'square',
  sides : 4
}

//accessing the properties using square brackets
console.log("Name is:", shape['name']) //using square brackets to access "name"
console.log("Number of sides are:", shape['sides']) //using square brackets to access "sides"

//creating an object named employee
var employee = {
  //defining properties of the object
  //setting data values

  //value of "name" is another object
  name : {
    firstName: 'Joe',
    lastName : 'Adams'
  }
}

//accessing the first name
console.log("First name is:", employee['name']['firstName'])

//accessing the last name
console.log("Last name is:", employee['name']['lastName'])
```

As seen above, the method to access a property is similar to how you access values in an array. In the case of nested objects, we access properties in the same way that a multidimensional array is accessed.

Iterating over Object Properties

The `for..in` loop present in JavaScript can be used in order to iterate over object *properties*.

Syntax

The syntax of the `for..in` loop is as follows:

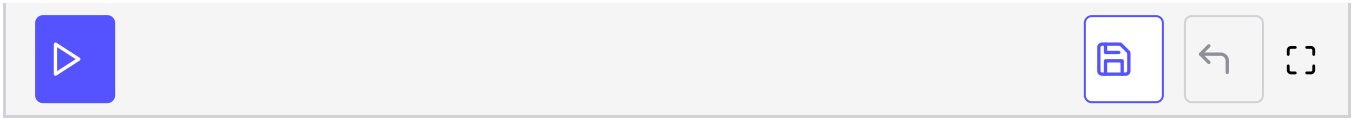
```
for (iteratorName in objectName)
```

Example

Let's take a look at an example implementing a `for..in` loop.

```
//creating an object named employee
var employee = {
  firstName: 'Joe',
  lastName : 'Adams',
  age: 23,
  sex: 'male',
  designation: 'chef'
}

//using for..in loop to iterate over the properties
for (x in employee){
  console.log(x)
}
```



In the above code, `x` is the iterator that will iterate over the *properties* one by one and display the property names on the console.

Calling Functions

The function property inside an object can be called using the dot operator as well as square brackets.

Syntax

The syntax for calling a method using dot notation is as follows:

```
objectName.functionName()
```

The syntax for calling a function using square brackets is as follows:

```
objectName['functionName']()
```

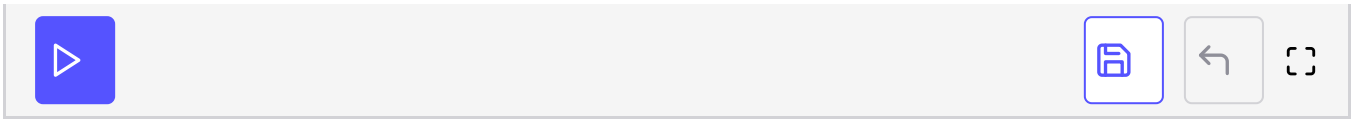
Example

Let's take a look at an example:

```
//creating an object named shape

var shape = {
  //defining properties of the object
  //function to display a message
  displaySides() {
    console.log("Square has 4 sides")
  },
  displayName() {
  console.log("Shape is Square")
  }
}

//calling function using dot operator
shape.displayName()
//calling function using square brackets
shape['displayName']()
//calling function using dot operator
shape.displaySides()
//calling function using square brackets
shape['displaySides']()
```



Now that you know how to access values in an object, let's discuss how to set the values in the next lesson.