DECT-KS© TOOLS

Web Engineering Front-end Pt. 3

6. JavaScript: Objects and Arrays

erb 僱員再培訓局 Employees Retraining Board
SCOPE School of Continuing and Professional Education 專業進修學院 香港城市大學 City University of Hong Kong
KRYSTAL INSTITUTE DIGITAL ECONOMY CORE TECHNOLOGY

Contents

1. Objects
2. Using Objects
3. Arrays
4. Using Arrays

6.1 Objects

What are objects?

JavaScript is an object-oriented programming language, which means almost everything in JS is an object.

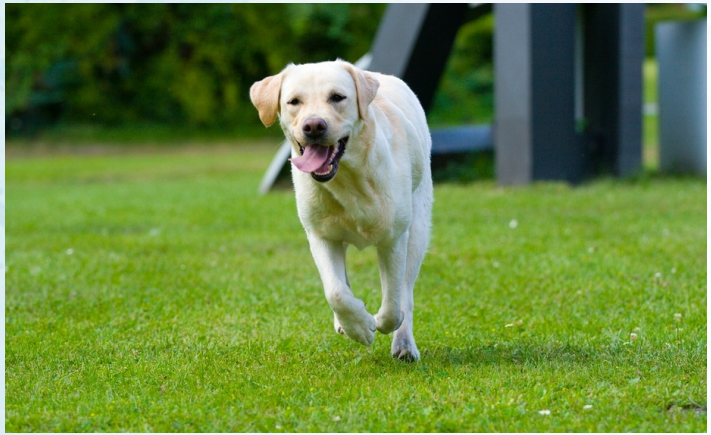
e.g. Strings, numbers, Booleans can all be objects.

What are objects?

Objects are made out of **properties** and **methods**.

Note: We will cover methods in a later topic.

Dog object



Properties:

- name: Marley
- gender: Male
- age: 2
- breed: Labrador retriever

Object properties

Object properties are made of (key: pair) values.

e.g.

name: “Marley”

Object literal

The first and easiest way to create objects is to use an object literal, where you create and define the object in one statement.

Object literal example

```
var dog = {
  name: “Marley”,
  gender: “Male”,
  age: 2,
  breed: “Labrador retriever”
};
```

new keyword

You can also use the new keyword and Object() method to create an empty object, then fill the object with variables.

new keyword example

```
var dog = new Object();
dog.name = “Marley”;
dog.gender = “Male”;
dog.age = 2;
dog.breed = “Labrador retriever”;
```

Object literal vs new keyword

Both methods do the exact same thing. For performance and readability, use the object literal method.

Practice: Objects

Try creating some objects with properties based on some real-life “things”.

e.g.

Animals, cars, locations…

DECT-KS©
TOOLS

6.2 Using Objects

Accessing object properties

object.property

or

object[“property”]

Accessing object properties

```
dog.gender; // “Male”

or

dog[“gender”]; // “Male”
```

Accessing object properties

To view the entire object, simply call the name of the object.

e.g.

```
dog; // {name: "Marley", gender: "Male", age: 2, breed:
"Labrador retriever"}
```

Modifying object properties

Object properties are mutable, which means they can be changed.

Modifying object properties

```
dog.name; // “Marley”

dog.name = “Rex”;

dog.name; // “Rex”
```

Removing object properties

Use the delete keyword to remove a property from an object.

e.g.

```
delete dog.breed;

dog; // {name: "Marley", gender: "Male“, age: 2}
```

Practice: Using objects

Spend some time practicing accessing object properties, adding properties, and deleting properties on the objects you created in the previous exercise.

6.3 Arrays

What are arrays?

Arrays are a special kind of object. They can be used to store multiple values into a single variable.

Why use arrays?

```
var student1 = “Chris”;
var student2 = “Kim”;
var student3 = “Ben”;
var student4 = “Matt”;
```

Why use arrays?

var students = [“Chris”, “Kim”, “Ben”, “Matt”]

Which is more efficient?

Arrays can be useful when you want to store a list of similar items

Arrays vs Objects

Arrays: [ ]

Objects: { }

Objects are filled with key: pair values, while arrays just contain values.

Note: typeof(students); // “Object”

Array literals

Similar to objects, the easiest way to create an array is using an array literal.

Array literals syntax

```
var arrayName = [item1, item2, ...];
or
var arrayName = [
  item1,
  item 2 ...
];
```

new keyword

You can use the new keyword to create arrays, but again it is not recommended.

```
var arrayName = new Array(item1, item2);
```

Practice: Arrays

Spend some time creating arrays. Try thinking of values that would make sense to put together into a single variable.

6.4 Using Arrays

Accessing array elements

Array elements are accessed using their index number.

e.g.

students[2]; // “Ben”

Accessing array elements

Remember when we counted the characters in a string?

Array indexes also start with 0.

Accessing array elements

```
students[0]; // “Chris”

students[1]; // “Kim”

students[2]; // “Ben”

students[3]; // “Matt”
```

.length

Use the .length property the return the number of elements in an array.

e.g.

```
students.length; // 4
```

.push()

The .push() method adds a new element to the end of the array.

.push()

```
students.push(“Tom”);

students; // [“Chris”, “Kim”, “Ben”, “Matt”, “Tom”]
```

Modifying array elements

Changing elements in arrays is just like changing properties in objects.

```
students[1] = 20

students[3] = false

students; // [“Chris”, 20, “Ben”, false]
```

Deleting array elements

Using the delete keyword will change the value of the element to undefined. It won't actually remove the element from the array.

Deleting array elements

```
delete students[0];

students; // [empty, “Kim”, “Ben”, “Matt”]

typeof(students[0]); // undefined
```

.pop()

The .pop() method removes the last element of an array.

.pop()

```
students.pop();

students; // [“Chris”, “Kim”, “Ben”]
```

.shift()

The .shift() method removes the first element of an array and shifts the other elements to a lower index.

.shift()

```
students.shift();

students; // [“Kim”, “Ben”, “Matt”]
```

.splice()

The .splice() method can be used to add new elements into the middle of an array.

.splice() syntax

array.splice(x, y, item1, item2, ...)

x: The position where new elements should be added

y: How many elements should be removed

item1, item2, ...: The new elements to be added

.splice() syntax

```
students.splice(1, 3, “Tom”, “Tim”, “Brad”);

students; // [“Chris”, “Tom”, “Tim”, “Brad”)
```

Practice: Using Arrays

Spend some time practicing accessing and modifying array elements using what you just learned on the arrays you created in the previous exercise.

DECT-KS©
TOOLS

The End

Reference 1: tutorialspoint JavaScript – Objects Overview

https://www.tutorialspoint.com/javascript/javascript_objects.htm

Reference 2: W3 Schools JavaScript Tutorial https://www.w3schools.com/js/default.asp

Reference 3: Labrador (Slide 6) https://www.flickr.com/photos/23807781@N06/3798577491