COURSEWARE



# Objects, Arrays + JSON



## Contents



## Overview

To explore Object, Arrays, and JSON object in JavaScript.

### Objects

An object can be defined as an unordered collection of related data, of primitive or reference types in the form of `key:value` pairs.
An object, is a reference data type.
Variables that are assigned a reference value are given a reference or a pointer to that value.
That reference or pointer points to the location in memory where the object is stored. **The variables don't actually store the value**.

### Arrays

Arrays holds a set of related data; for example students in a class.

Arrays in JavaScript have some idiosyncrasies :

- They can be resized at any time
- They index at 0
  - So Array(3) would have elements with indexes 0, 1 and 2
- They can be sparsely filled
- Unassigned parts of an array are undefined
- They can be created in short hand using just square brackets

### JSON

JSON stands for **J**ava **S**cript **O**bject **N**otation

Lightweight data-interchange format:

- Compared to XML

Simple format:

- Easy for humans to read and write
- Easy for machines to parse and generate

JSON is a text format:

- Programming language independent
- Conventions familiar to programmers of the C-family of languages, including C# and JavaScript

# Tutorial

## How to create objects

```
let car = new Object();
car["make"] = "Audi";
car["model"] = "A5";
car["carReg"] = "W8M8";

console.log(car) // shows all attributes
console.log(car.make); //show the value of make
console.log(car.model); //show the value of model
console.log(car.carReg); //show the value of carReg

//The object can have new properties added at any time. Known as an expando
property.

car.numberOfDoors = 4;
console.log(car.numberOfDoors); //Shows the value referring to number of doors.
```

How to create objects using literal notation

```
//a lot simpler and quicker to declare objects

let car = { make: "Audi", model: "A5", carReg: "W8M8" };
```

Using literal notation to create complex arrays of objects.

```
 let carPark = [
    { make: "Audi", model: "A5", carReg: "AB12CDE" },
    { make: "hyundai", model: "i30", carReg: "AB11CDE" }
]
console.log(carPark);
```

▶ How to loop through arrays of objects

## How to create arrays

```
let a = Array(); // an empty array called a
let b = Array(10); //array containing 10 empty spaces called b
let c = Array("Tom","Dick","Harry"); // array containing 3 elements called c
// short hand for creating arrays
let cShortHand = ["Tom","Dick","Harry"]; //shorthand using [] around each elment
and seperated by a comma.
let d = [1,2,3,4,5];
```

Important things to note about arrays:

- They can be resized at any time
- They index at 0
  - So Array(3) would have elements with indexes 0, 1 and 2
- They can be sparsely filled
- Unassigned parts of an array are `undefined`
- They can be created in short hand using just square brackets

How to access arrays

```
let classRoom = ["","","",""];
classRoom[0] = "John";
classRoom[3] = "Simran";

console.log(classRoom[0]); // will return John
console.log(classRoom[2]); // will return undefined
console.log(classRoom[3]); // will return Simran.
```

▶ How to loop through arrays

## Array object methods

Methods can be used with arrays in order to carry out certain functions.

| Method | Description |
| --- | --- |
| sort() | Sorts the array using string comparisons by default. |
| reverse() | reverses the elements in an array. |
| join() | Joins all the elements of the array into one string, using the supplied separator or a comma |

```
let d =["apple","strawberries","banana", "grapes", "pear"];

console.log(d); // ["apple", "strawberries", "banana", "grapes", "pear"]
console.log(d.sort()); // ["apple", "banana", "grapes", "pear", "strawberries"]
console.log(d.reverse());//["strawberries", "pear", "grapes", "banana", "apple"]
console.log(d.join()); //strawberries,pear,grapes,banana,apple

e = d.join("-");
console.log(e);// strawberries-pear-grapes-banana-apple
```

| Method | Description |
| --- | --- |
| push() | adds a new element to the end of the arrays |
| pop() | removes the last element from the end of the arrays |

```
    console.log(d.push("lemon"));
    console.log(d);//["strawberries", "pear", "grapes", "banana", "apple",
"lemon"]
    console.log(d.pop());
    console.log(d);//["strawberries", "pear", "grapes", "banana", "apple"]
```

| Method | Description |
| --- | --- |
| unshift() | adds a new element to the beginning of the arrays |
| shift() | remove the first element from the beginning of the array |

```
    console.log(d.unshift("kiwi"));
    console.log(d); //["kiwi", "strawberries", "pear", "grapes", "banana",
"apple"]
    console.log(d.shift());
    console.log(d); //["strawberries", "pear", "grapes", "banana", "apple"]
```

```
let myArray = [1,2,3,4]; //declare an array
//create a temp variable called "eachElement"
//iterate though "myArray" starting at index 0
//finally log the value of "eachElement"
for (let eachElement of myArray) {
    console.log(eachElement);
}
```

## How to JSON objects

JSON is a subset of the object literal notation of JavaScript.
It can be used in the JavaScript language with no problems.

- A collection of name/value pairs
  - Realised as an object (associative array)
    An ordered list of values
  - Realised as an array
- JSON object
  - Unordered set of name/value pairs
  - Begins with { (left brace) and ends with } (right brace)
  - Each name followed by a : (colon)
  - Name/Value pairs separated by a , (comma)

```
let myJSONObject = {
    "searchResults": [
        {
            "productName": "Aniseed Syrup",
            "unitPrice": 10
        },
        {
            "productName": "Alice Mutton",
            "unitPrice": 39
        }
    ]
};
```

The JSON object is globally available

```
//The parse method takes a string and parses it into JavaScript objects
let obj = JSON.parse('{"name":"Adrian"}');
console.log(obj.name); //returns Adrian
//The stringify method takes JavaScript objects and returns a string
let str = JSON.stringify({ name: "John" });
```

There are a series of overloaded methods for the type:

| Method | Description |
| --- | --- |
| JSON.parse( text ) | Converts a serialised JSON string into a JavaScript object. |
| JSON.parse( text, translate ) | Uses a translation function to convert values or remove them entirely. |
| JSON.stringify( obj ) | Converts an object into a serialised JSON string. |
| JSON.stringify( obj, ["white", "list"]) | Serialises only a specific white list of properties. |
| JSON.stringify( obj, translate ) | Serialises the object using a translation function. |
| JSON.stringify( obj, null, 2 ) | Adds the specified number of spaces to the output, printing it evenly. |

## Exercises

1. Create an object called `darthVader` with the keys `allegiance`, `weapon` and `sith` and the values of `empire`, `lightsabre` and `true`. Finally log `darthVader`

   ▶ Solutions
2. Create the following log statements using the `darthVader` Object

- Darth Vader's allegiance is to the `Empire`;
- Darth Vader's weapon of choice is a `lightsabre`;
- Darth Vader is a sith? `true`;
- Darth Vader is a Jedi? `false`;

▶ Solutions

3. 1. Create an array with the name `myArray` with 2 elements `hello`,`everyone.`.
   2. Next print the length of the array
   3. Next use the `push()` method to add 3 elements to the array
   4. Next print the length of the array
   5. Next use `shift()` to remove an element
   6. Finally print the contents of the array using a for of loop.

   ▶ Solutions