Chapter 05 : Objects

Object Literals

- An object in JavaScript is a self-contained set of related values and functions.
- If a property's value is a function, it is known as a **method**.
- Here is an example of an object literal that describes the Man of Steel:

```
const superman = {
name: 'Superman',
'real name': 'Clark Kent',
height: 75,
weight: 235,
hero: true,
villain: false,
allies: ['Batman','Supergirl','Superboy'],
fly() {
return 'Up, up and away!';
}
};
```

- All objects are mutable at any time when a program is running.

Creating Objects

```
const spiderman = {};
const spiderman = new Object();
```

- ES6 provided a shorthand method of creating objects if a property key is the same as a variable name that the property value is assigned to:

```
// long way
const ironMan = ( name: name, realName: realName };
// short ES6 way
const ironMan = { name, realName };
```

Accessing Properties

- You can access the properties of an object using the dot notation: superman.name
- You can also access an object's properties using bracket notation: superman['name']

- If you try to access a property that doesn't exist, undefined will be returned

Computed Properties

- JavaScript code can be placed inside square brackets and the property key will be the return value of that code.

```
const hulk = { name: 'Hulk', ['catch' + 'Phrase']: 'Hulk Smash!' };
```

- The new Symbol date type can also be used as a computed property key:

```
const supergirl = { [name]: 'Supergirl' };
```

- The symbols used for property keys are not limited to being used by only one object - they can be reused by any other object:
- To call an object's method we can also use dot or bracket notation:

```
superman.fly()
superman['fly']()
```

Checking if Properties or Methods Exist

- The in operator can be used to check whether an object has a particular property:

```
'city' in superman
```

- Alternatively, you could also check to see if the property or method doesn't return *undefined*
- Another way is to use the hasOwnProperty() method:

```
superman.hasOwnProperty('city');
```

- We can loop through all of an object's properties and methods by using a for in loop.

```
for(const key in superman) {
    console.log(key + ": " + superman[key]);
}
```

- The Object.keys() method will return an array of all the keys of any object that is provided as an argument:

```
for(const key of Object.keys(superman)) {
  console.log(key);
}
```

- ES2017 also adds some the Object.values() that works in the same way, but returns an array of all the object's value:

```
for(const value of Object.values(superman)) {
console.log(value);
```

```
}
```

- Object.entries() is also part of ES2017 and returns an array of key-value pairs.These key-value pairs are returned in arrays, but they can be destructured and accessed individually:

```
for(const [key,value] of Object.entries(superman)) {
    console.log(`${key}: ${value}`);
}
```

Adding Properties

- New properties and methods can be added to objects at any time in a program.
- This is done by simply assigning a value to the new property.
- Any property can be removed from an object using the delete operator: delete superman.fly
- You can change the value of an object's properties at any time using assignment.

- It's even possible for an object to contain other objects. These are known as nested objects.
  - An object literal can be passed as a parameter to a function.
  - The keyword this refers to the object that it is within.
  - -object literal pattern to create a namespace for groups of related functions.

JSON

- JavaScript Object Notation, or JSON is an extremely popular lightweight data-storage format that is used by a large number of services for data serialization and configuration.
- JSON is a string representation of the object literal notation that we have just seen.
- The parse() method takes a string of data in JSON format and returns a JavaScript object.
- The stringify() method does the opposite.

The Math Object

- Math.PI : The ratio of the circumference and diameter of a circle.
- Math.SQRT2 : The square root of 2
- Math.SQRT1_2 : The reciprocal of the square root of 2
- Math.E : Euler's constant
- Math.LN2 : The natural logarithm of 2
- Math.LN10 : The natural logarithm of 10
- Math.LOG2E : Log base 2 of Euler's constant
- Math.LOG10E : Log base 10 of Euler's constant
- Math.abs(3) : method returns the absolute value of a number.
- Math.ceil(): round a number *up* to the next integer
- Math.floor(): method will round a number *down* to the next integer
- Math.round(): round a number to the *nearest* integer
- Math.trunc(): returns the integer-part of a number