Data processing

Prepared by Alex Sobolewski

Hello!

My name is Alex Sobolewski

I have been working in IT as a programmer since 2013. My specialization is JavaScript and Java based solutions. I am strong adherent of Clean Code and good design architecture. Currently i am working as Full Stack Engineer.

1.
Objects in JS

Objects

What is an object?

In JS an object is a logic entity, which connects logically related properties. Consider this object literal:

```
let cat = {
        size: “average”,
        color: “black”,
        name: “Kitty”
};
```

An object cat was declared. It has logical(for cat) properties. Properties of this object just like variables, but isolated to this cat object.

Objects
What is an object?

We also can use object in object:

```
let cat = {
        size: “average”,
        color: “black”,
        name: “Kitty”,
        favoriteThings: {
                toys: [“Mouse”, “Big mouse”],
                meals: [“Whiskas”]
        }
};
```

As you can see - we are not restricted to using only primitive properties, we also can use objects in objects, objects in arrays, arrays in objects and etc.

Objects

What is an object?

Objects have not only properties, but methods too. Consider this:

```
let cat = {
        …,
        meow: function(){
                console.log(“I am a cat! Meow!”)
        }
};
```

Besides properties, an object can have many methods. Methods are like actions related to this logical object.

Objects

Declaration of object

We can declare an object in two ways:

- let cat = new Object();
- let cat = {};

You are free to choose how to declare it.

Objects
Working with object properties

In order to read properties of an object we can do two things:

- Read it directly: **cat.color;**
- Read it through braces: **cat['color'];**

Both ways are correct and sometimes the second one is the only way to read a property.

Consider this:
cat.favoriteThings['toys'][1] - we can use chain of reading commands.

Objects
Working with object properties

In order to write a value to a property of an object we can do two things:

- Write it directly: **cat.color = “Red”;**
- Write it through braces: **cat[‘color’] = “Red”;**

Both ways are correct and sometimes the second one is the only way to change a property.

We also can create new property by this method:

cat.tailColor = ‘Red’; // tailColor was’t in cat object, but it will be added now

Objects
Working with object methods

In order to call methods of an object we can do two things:

- Call it directly: **cat.meow();**
- Call it through braces: **cat[‘meow’]();**

Both ways are correct and sometimes the second one is the only way to call a method.

Objects

Objects pitfalls

The most common problem with objects is that they are passed by reference and often it creates some misunderstandings. Consider this:

```
let cat = {...};
let anotherCat = cat;
cat.color = “Blue”;
anotherCat.color; // will be blue as well, because it is the same objects
```

Objects
For in

We can iterate over an object with for … in statement:

```
const  obj = {name: 'Bob', age: 33};

for(const prop in obj){
        console.log(obj[prop]);
}
```

Objects
Object's .entries() method

The .entries() method represents property and its value as 2-element array, so the object is represented as array of arrays:

const obj = {name: 'Bob', age: 33};
Object.entries(obj) // [['name', 'Bob'], ['age', 33]]

Objects

Object's .keys() and .values() methods

We can access object properties(as names) directly by .keys() method:
const obj = {name: 'Bob', age: 33};
Object.keys(obj) // ['name', 'age'];

We can access object values(as values) directly by .values() method:
Object.values(obj) // ['Bob', '33'];

Objects

Object's string representation

Sometimes we want to keep our objects like string, and not like object.
For this we have few useful methods:

const x = {name: 'Kasia', age: 22};
JSON.stringify(x); // '{name: "Kasia", age: 22}'

In order to revert this change we use another JSON's method:

JSON.parse('{name: "Kasia", age: 22}'); // result in x object

Objects
Math object

Math object - is special built-in object in JS. It handles math operations like cos, tang, floor, round and etc.

Math.floor(); // rounds down, 5.5 becames 5
Math.round(); // rounds to up if the value > 0.5 and rounds down if the value < 0.5

Objects
Math.random()

In JS there is very helpful Math object, which has plenety of useful functions. One of them - .random() function:

Math.random(); // 0.5689146912

We can create random numbers based on random result:

Number(Math.random() * 100).toPrecision(4);

Here .toPrecision() method will cut out unnecessary precision.

Objects
Date object

In JS there is a built-in Date object which handles dates.

```
Date.now();  // 5689146912 in ms

new Date(); // new date object, which points to time in the moment of date object creation

someDate.toUTCString(); // Wed, 14 Jun 2017 07:00:00 GMT
const time = Date.parse("11/30/2011");
(new Date(time)).toUTCString(); // Tue, 29 Nov 2011 23:00:00 GMT
```

2.
The end