SOFT CLUB

JAVA SCRIPT

LECTURE 5

SOFT CLUB

TABLE OF CONTENTS

SOFT CLUB

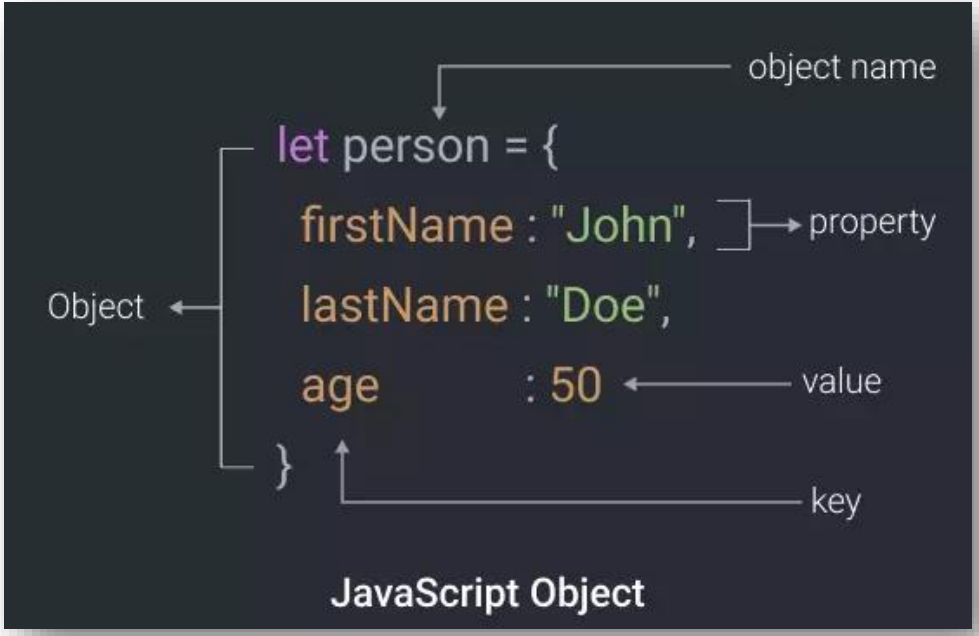
What is **Object** in **JavaScript** ?

WHAT IS A OBJECT IN JAVA SCRIPT ?

In JavaScript, an object is a standalone entity, with properties and type.
Compare it with a cup, for example. A cup is an object, with properties. A cup has a color, a design, weight, a material it is made of, etc. The same way, JavaScript objects can have properties, which define their characteristics.
Everything is an object in JavaScript.

CREATE OBJECT

JavaScript object is a nonprimitive data-type that allows you to store multiple collections of data



METHODS OBJECT

SOFT CLUB

Object.entries()

```
const obj = { name: "Adam", age: 20,};
console.log(Object.entries(obj));
// Output: [ [ 'name', 'Adam' ], [ 'age', 20 ] ]
```

Object. keys()

```
const obj = { name: "Adam", age: 20,};
console.log(Object.keys(obj));
// Output:  [ 'name', 'age' ]
```

Object. values()

```
const obj = { name: "Adam", age: 20,};
console.log(Object.values(obj));
// Output:  [ 'Adam', 20 ]
```

{ }

What is Destructuring and Spread in JavaScript ?

DESTRUCTURING AND SPREAD IN OBJECT

The destructuring assignment syntax is a JavaScript expression that makes it possible to unpack properties from object, into distinct variables

```
const person = {
    name: 'Sara',
    age: 25,
    gender: 'female'
}

let name = person.name;
let {age,gender} = person
let cloneObject={...person}
console.log(cloneObject)
//{ name: 'Sara', age: 25, gender: 'female' }
console.log(cloneObject==person) // false
console.log(name); // Sara
console.log(age); // 25
console.log(gender); // female
```

assigning object attributes to variables

1 **Before ES6:**

2 **From ES6:**

3 **Clone Object (spread):**

SOFT CLUB

What is keyword “ this ” in JavaScript ?

HOW THE THIS KEYWORD WORKS

SOFT CLUB

👉 **`this` keyword/variable:** Special variable that is created for every execution context (every function). Takes the value of (points to) the "owner" of the function in which the `this` keyword is used.

☝️ `this` is **NOT** static. It depends on **how** the function is called, and its value is only assigned when the function **is actually called**.

The this keyword refers to different objects depending on how it is used:

In an object method, `this` refers to the **object**.
Alone, `this` refers to the **global object**.
In a function, `this` refers to the **global object**.
In a function, in strict mode, `this` is `undefined`.

this is **NOT** a variable. It is a **keyword**. You cannot change the value of **this**.

👉 Method example:

```
const jonas = {
  name: 'Jonas',
  year: 1989,
  calcAge: function() {
    return 2037 - this.year;
  }
};
jonas.calcAge(); // 48
```

calcAge is method

jonas

1989

Way better than using `jonas.year`!

SOFT CLUB

What is **new Date()** in **JavaScript** ?

DATA AND TIME

In JavaScript, date and time are represented by the Date object. The Date object provides the date and time information and also provides various methods. A JavaScript date defines the EcmaScript epoch that represents milliseconds since 1 January 1970 UTC. This date and time is the same as the UNIX epoch (predominant base value for computer-recorded date and time values).

Creating Date Objects

There are four ways to create a date object.

- new Date()
- new Date(milliseconds)
- new Date(Date string)
- new Date(year, month, day, hours, minutes, seconds, milliseconds)

DATA AND TIME

new Date()

You can create a date object using the new Date() constructor. For example,

```
1  // shows current date and time
2  const timeNow = new Date();
3  console.log(timeNow);
```

```
node /tmp/5NmvGiZKLA.js
2022-02-16T10:03:43.408Z
```

new Date(milliseconds)

A Date object contains a number representing milliseconds since January 1, 1970 UTC. New Date(milliseconds) Creates a new Date object by adding milliseconds to the zero time. For example,

```
1  const time1 = new Date(0);
2
3  // epoch time
4  console.log(time1); // Thu Jan 01 1970 05:30:00
5
6  // 100000000000 milliseconds after the epoch time
7  const time2 = new Date(100000000000)
8  console.log(time2); // Sat Mar 03 1973 15:16:40
```

```
node /tmp/5NmvGiZKLA.js
1970-01-01T00:00:00.000Z

1973-03-03T09:46:40.000Z
```

new Date (date string)

New Date - (date string) creates a new date object from the date string.In JavaScript, there are generally three date input formats. ISO Date Formats You can create a date object by passing ISO date formats. For example,

```
// ISO Date(International Standard)
const date = new Date("2022-02-16");

// the result date will be according to UTC
console.log(date);
// You can also pass only the year and month or only the year. For example,
const date2 = new Date("2022-02");
console.log(date2); //

const date3 = new Date("2022");
console.log(date3);
// you can also pass specific time to ISO dates.
const date4 = new Date("2001-12-09T11:00:00Z");
console.log(date4);
```

```
node /tmp/5NmvGiZKLA.js
2022-02-16T00:00:00.000Z

2022-02-01T00:00:00.000Z

2022-01-01T00:00:00.000Z

2001-12-09T11:00:00.000Z
```

DATA AND TIME

new Date (year, month, day, hours, minutes, seconds, milliseconds)

new Date(year, month,...) creates a new date object with the specified date and time. For example,

```
1  // new Date(year, month,...) creates a new date object by passing specific date and time. exsample
2  const time1 = new Date(2020, 1, 20, 4, 12, 11, 0);
3  console.log(time1);
4  // If four numbers are passed, it represents year, month, day and hours. For example,
5  const time2 = new Date(2020, 1, 20, 4);
6  console.log(time2); // Thu Feb 20 2020 04:00:00
7
8  const time3 = new Date(2020, 1);
9  console.log(time3);
```

```
node /tmp/5NmvGiZKLA.js
2020-02-20T04:12:11.000Z
2020-02-20T04:00:00.000Z
2020-02-01T00:00:00.000Z
```

Note: If you pass only one argument, it will count as milliseconds. Therefore, to use this date format, you must pass two arguments. In JavaScript, months are counted from 0 to 11. January is 0 and December is 11.

DATA AND TIME

Method	Description
now()	Returns the numeric value corresponding to the current time (the number of milliseconds elapsed since January 1, 1970 00:00:00 UTC)
getFullYear()	Gets the year according to local time
getMonth()	Gets the month, from 0 to 11 according to local time
getDate()	Gets the day of the month (1–31) according to local time
getDay()	Gets the day of the week (0-6) according to local time
getHours()	Gets the hour from 0 to 23 according to local time
getMinutes	Gets the minute from 0 to 59 according to local time
getUTCDate()	Gets the day of the month (1–31) according to universal time
setFullYear()	Sets the full year according to local time
setMonth()	Sets the month according to local time
setDate()	Sets the day of the month according to local time
setUTCDate()	Sets the day of the month according to universal time

DATA AND TIME

now()

Date.now() returns the number of milliseconds since January 1, 1970.

```
1
2  var n = Date.now();
3  console.log(n);
```

```
node /tmp/dpPCeMf2gv.js
1645012052687
```

getFullYear()

getFullYear() returns the full year of a date (4 digits).

```
1  const time = new Date();
2  let year = d.getFullYear();
3  console.log(year)
```

```
node /tmp/dpPCeMf2gv.js
2022
```

getMonth()

getMonth() returns the month (0 to 11) of the date. January = 0, February = 1, ..

```
1  const d = new Date();
2  let month = d.getMonth();
3  console.log(month)
```

```
node /tmp/dpPCeMf2gv.js
1
```

```
1  const month = ["January","February","March","April","May","June","July","August","September"
       ,"October","November","December"];
2  const d = new Date();
3  let name = month[d.getMonth()];
4  console.log(name)
```

```
node /tmp/dpPCeMf2gv.js
February
```

DATA AND TIME

SOFT CLUB

getDate()

The getDate() method returns the day of the month (1 to 31) of the date.

```
1  const d = new Date();
2  let day = d.getDate();
3  console.log(day)
```

```
node /tmp/dpPCeMf2gv.js
16
```

getDay()

The getDay() method returns the day of the week (0 to 6) of the date. Sunday = 0, Monday = 1,

```
1  let day= new Date();
2  console.log(day.getDay())
```

```
node /tmp/dpPCeMf2gv.js
3
```

getHours()

getHours() returns the hours (0 to 23) of a date.

```
1  let day= new Date();
2  console.log(day.getHours())
```

```
node /tmp/dpPCeMf2gv.js
12
```

getMinutes

getMinutes() returns the minutes (0 to 59) of a date.

```
1  let day= new Date();
2  console.log(day.getMinutes())
```

```
node /tmp/dpPCeMf2gv.js
34
```

We will show you a minute

```
const d = new Date("July 21, 1983 01:15:00");
let minutes = d.getMinutes();
console.log(minutes)
```

```
node /tmp/dpPCeMf2gv.js
15
```

DATA AND TIME

setDate()

setDate() sets the day month of the date.

```
1  // бо сану рӯзу моҳ ва соат,,,,
2  const d = new Date();
3  d.setDate(15);
4  console.log(d);
5  // бо милисония
6  const d2 = new Date();
7  console.log(d2.setDate(15));
8  // милисекунда ба сану рӯзу моҳ,,,
9  let a = new Date(d2);
10 console.log(a.toString());
```

```
node /tmp/dL3BIJT1Ak.js
2022-02-15T10:31:42.089Z


1644921102093

Tue Feb 15 2022 10:31:42 GMT+0000 (GMT)
```

DATA AND TIME

setMonth()

The setMonth() method sets the month of the date object.

This method can also be used to set the day of the month.

```
1  // бо сану ҳозу мох ва соат,,,,                    node /tmp/dL3BIJTiAk.js
2  const d = new Date();                              2022-05-17T10:26:07.644Z
3  d.setMonth(4);
4  console.log(d);
5  // бо милисония
6  const d2 = new Date();
7  console.log(d2.setMonth(4));                       1652783167649
```

DATA AND TIME

setFullYear()

setFullYear() sets the year of the date.
setFullYear() can also set the month and day.

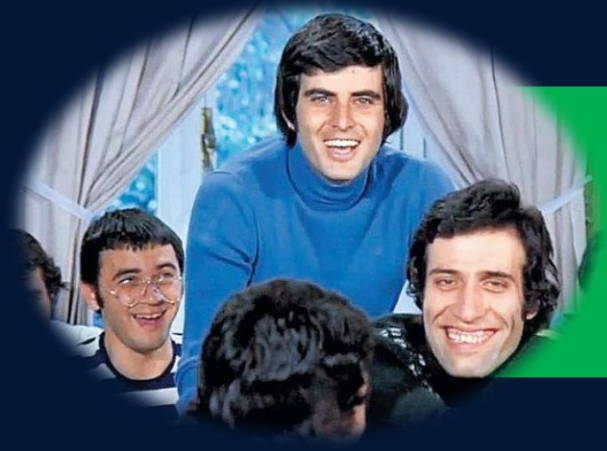
```
1  //Sets the full year according to local time
2  const day = new Date();
3  console.log(day.setFullYear(2020));
4  // Милисикунда ба сату дузу моҳ гардондем
5  let a= new Date(day)
6  console.log(a.toString());
7  // Set the date to six months ago:
8  const d = new Date();
9  d.setFullYear(d.getFullYear(), d.getMonth() - 6);
10 console.log(d)
```

```
node /tmp/dL3BIJTiAk.js
1581934679119

Mon Feb 17 2020 10:17:59 GMT+0000 (GMT)

2021-08-17T10:17:59.125Z
```

SOFT CLUB

Thanks!

Be happy and Smile



THE END

LECTURE 5