

Date() ,Math , Random And SetInterval()

```
var x = Date.parse('March 15, 2000')
```

```
var para = new Date(x)
```

```
new Temporal.PlainDate(2024, 7, 26);
```

```
Temporal.PlainDate.from('2024-07-26');
```

```
// both return a PlainDate object that represents 26th July 2024
```

```
new Temporal.PlainTime(20, 24, 0);
```

```
Temporal.PlainTime.from('20:24:00');
```

```
// both return a PlainTime object of 20:24
```

```
const valentinesDay = Temporal.PlainMonthDay.from({ month: 2, day: 14 });
```

```
const march = Temporal.PlainYearMonth.from({ month: 3, year: 2024 });
```

```
var _day = new Date()
```

```
_day.setDate(-6550)
```

```
const d = new Date("03/25/2015");
```

```
var _day = new Date()
```

```
_day.setFullYear(2050,10,20)
```

```
const date = Date.now(); => Return the current date/time in milliseconds since January 1, 1970
```

getFullYear()	Get year as a four digit number (yyyy)
getMonth()	Get month as a number (0-11)
getDate()	Get day as a number (1-31)
getDay()	Get weekday as a number (0-6)
getHours()	Get hour (0-23)
getMinutes()	Get minute (0-59)
getSeconds()	Get second (0-59)
getMilliseconds()	Get millisecond (0-999)
getTime()	Get time (milliseconds since January 1, 1970)

```
Math.E      // returns Euler's number
Math.PI     // returns PI
Math.SQRT2  // returns the square root of 2
Math.SQRT1_2 // returns the square root of 1/2
Math.LN2    // returns the natural logarithm of 2
Math.LN10   // returns the natural logarithm of 10
Math.LOG2E  // returns base 2 logarithm of E
Math.LOG10E // returns base 10 logarithm of E
```

JavaScript Math Object

```
Math.PI
Math.E
Math.round(2.3)
Math.ceil(4.1) => round to up
Math.floor(4.7) => round to floor
Math.trunc(4.999) => returns the integer
Math.sign(545) => returns if x is negative, null or positive
Math.sqrt(64)
Math.abs(-4.4)
Math.sin(90 * Math.PI / 180)
Math.min(0, 150, 30, 20, -8, -200)
Math.max(0, 150, 30, 20, -8, -200)
```

example for random:

```
<button onclick="document.getElementById('demo').innerHTML = getRndInteger(5,10)">Click Me</button>
```

```
function getRndInteger(min, max) {  
    return Math.floor(Math.random() * (max - min)) + min;  
}
```

```
Boolean(10 > 9)
```

Object

An object, in object-oriented programming (OOP), is **an abstract data type created by a developer**. It can include multiple properties and methods and may even contain other objects. In most programming languages, objects are defined as classes. ... A simple example of an object may be a user account created for a website.

In JavaScript, almost "everything" is an object.

- Booleans can be objects (if defined with the **new** keyword)
- Numbers can be objects (if defined with the **new** keyword)
- Strings can be objects (if defined with the **new** keyword)
- Dates are always objects
- Maths are always objects
- Regular expressions are always objects
- Arrays are always objects
- Functions are always objects
- Objects are always objects

Syntax : `objectName.propertyName` or `objectName["propertyName"]`

```
const person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};
```

```
const person = {  
    firstName: "John",  
    lastName : "Doe",  
    id    : 5566  
};person.lastName or person["lastName"]
```

```
var person = new Object()  
  
person.name = 'hossein'  
  
person.lname = 'samiee'  
  
person.age = 15
```

```
var txt = "  
for(para in person){  
    txt = txt + person[para] + '<br>'  
}
```

```
delete Obj.name
```

```
var person = {  
    name : 'parsa',  
    lname : 'ghorbanian',  
    age : 50,  
    fullName : function(){  
        return this.name + " - " + this.lname  
    }  
}  
document.write(person.fullName())
```

```
var person = {  
    name : 'parsa',  
    lname : 'ghorbanian',  
    age : 50,  
}  
person.fullName = function(){  
    return this.name + " - " + this.lname  
}  
document.write(person.fullName())
```

```
var person = {  
    name : 'parsa',  
    lname : 'ghorbanian',  
    age : 50,  
}  
Object.defineProperty(person, "change", {  
    get :function(){this.age = 30}  
})  
person.change  
document.write(person.age)
```

```
function person(name, lname, age){
    this.firstName = name;
    this.lastName = lname;
    this.age = age
}
var fullName = new person('parsa', 'gh', 30)
document.write(fullName.firstName + " - " + fullName.age + " - " + fullName.lastName)
```

```
function person(name, lname) {
    this.name = name;
    this.lname = lname
}
person.prototype.age = 30
var myObjetc = new person('parsa', 'ghorbanian')
document.write(myObjetc.name + " - " + myObjetc.lname + " - " + myObjetc.age)
```

```
function person(name, lname) {
    this.name = name;
    this.lname = lname
}
person.prototype.fullname = function(){
    return this.name + " - " + this.lname
}
var myObjetc = new person('parsa', 'ghorbanian')
document.write(myObjetc.name + " - " + myObjetc.lname + " <hr> " + myObjetc.fullname())
```

```
var person = {
    name : 'ali',
    lname : 'aliani',
    age : 30
}
// change prop
Object.defineProperty(person, 'age', {value : 50})
Object.defineProperty(person, 'eyeColor', {value : 'brown'})
document.write(person.age + " - " + person.eyeColor)
```

```
var person = {  
  name : 'ali',  
  lname : 'aliani',  
  age : 30  
}  
  
document.write(Object.keys(person))
```

```
const person1 = {  
  fullName: function() {  
    return this.firstName + " " + this.lastName;  
  }  
}  
  
const person2 = {  
  firstName:"John",  
  lastName: "Doe",  
}  
  
let x = person1.fullName.call(person2);
```

```
const person = {  
  fullName: function() {  
    return this.firstName + " " + this.lastName;  
  }  
}  
  
const person1 = {  
  firstName:"John",  
  lastName: "Doe"  
}  
  
const person2 = {  
  firstName:"Mary",  
  lastName: "Doe"  
}  
  
document.getElementById("demo1").innerHTML = person.fullName.call(person1);  
document.getElementById("demo2").innerHTML = person.fullName.call(person2);  
.....
```

```
const person = {
  fullName: function(city, country) {
    return this.firstName + " " + this.lastName + ", " + city + ", " + country;
  }
}

const person1 = {
  firstName:"John",
  lastName: "Doe"
}

const person2 = {
  firstName:"Mary",
  lastName: "Doe"
}

document.getElementById("demo").innerHTML = person.fullName.call(person1, "tehran", "IR");
```

```
const person = {
  fullName: function() {
    return this.firstName + " " + this.lastName;
  }
}

const person1 = {
  firstName:"John",
  lastName: "Doe"
}

document.getElementById("demo").innerHTML = person.fullName.apply(person1);
```

.....

The call() method takes arguments separately.

The apply() method takes arguments as an array.

.....

```
const person = {
  fullName: function(city, country) {
    return this.firstName + " " + this.lastName + ", " + city + ", " + country;
  }
}

const person1 = {
  firstName:"John",
  lastName: "Doe"
}

person.fullName.apply(person1, ["tehran", "IR"]);
```

With the bind() method, an object can borrow a method from another object.

```
const person = {
  firstName:"John",
  lastName: "Doe",
  fullName: function () {
    return this.firstName + " " + this.lastName;
  }
}

const member = {
  firstName:"parsa",
  lastName: "gh",
}

let fullName = person.fullName.bind(member);    => “parsa gh”
```

```
.....

var person = {
  firstName: "John",
  lastName : "Doe",
  get fullName() {
    return this.firstName + " " + this.lastName;
  }
};

document.getElementById("demo").innerHTML = person.fullName;
```

```
.....

var person = {
  firstName: "John",
  lastName : "Doe",
  language : "NO",
  get lang() {
    return this.language;
  },
  set lang(value) {
    this.language = value;
  }
};

person.lang = "en";

document.getElementById("demo").innerHTML = person.lang;
```


prototype allows you to add new properties and methods to arrays.

prototype is a property available with all JavaScript objects.

```
function Person(first, last, age, eye) {  
    this.firstName = first;  
    this.lastName = last;  
    this.eyeColor = eye;  
}  
  
const myFather = new Person("John", "Doe", "blue");  
  
const myMother = new Person("Sally", "Rally", "green");
```

```
Person.prototype.nationality = "English";
```

```
document.getElementById("demo").innerHTML =  
  
"My father is " + myFather.nationality + "<br>" +  
  
"My mother is " + myMother.nationality;
```

Prototype

<https://trainingsitedesign.ir/%D9%BE%D8%B1%D9%88%D8%AA%D9%88%D8%AA%D8%A7%DB%8C%D9%BE%D9%87%D8%A7-%D8%AF%D8%B1-%D8%AC%D8%A7%D9%88%D8%A7%D8%A7%D8%B3%DA%A9%D8%B1%DB%8C%D9%BE%D8%AA/>

CSSOM => CSS Object Model

```
var MyTag = document.getElementById("mytag");
```

```
var x = MyTag.style.cssText = "text-align:center; color:blue;";
```

```
<h2 style=" text-align:center; color:blue; " id="mytag">parsa ghorbanian</h2>
```

```
var MyTag = document.getElementById("mytag");
```

```
var x = MyTag.style.cssText;
```

```
document.getElementById("Result").innerHTML = x;
```

```
var result = MyTag.style.length;
```

```
<style>
```

```
h2{  
    color: blue;}
```

```
p{  
color:red}
```

```
</style>
```

```
</head>
```

```
<body>
```

```
<button onclick="Get();">کلیک کنید</button>
```

```
<p id="Result"></p>
```

```
<script>
```

```
function Get(){
```

```
var MyStyle = document.styleSheets[0].rules[1].style;
```

```
var MyBlock = MyStyle.parentRule;
```

```
document.getElementById("Result").innerHTML = MyBlock.cssText;
```

```
}
```

```
</script>          =>p { color: red; }
```

.....

```
<button onclick="Set();">set</button>
```

```
<h2 id="mytag">testtttt</h2>
```

```
<script>
```

```
function Set(){
```

```
var MyTag = document.getElementById("mytag");
```

```
var New_Property_1 = MyTag.style.setProperty("background-color", "blue");
```

```
var New_Property_2 = MyTag.style.setProperty("color", "#fff");
```

```
var New_Property_3 = MyTag.style.setProperty("text-align", "center");
```

```
}
```

```
</script>
```

.....

```
<button onclick="Get();">get</button>
```

```
<h2 style="color:blue;" id="mytag">parsa ghorbanina</h2>
```

```
<p id="Result"></p>
```

```
<script>
```

```
function Get(){
```

```
var MyTag = document.getElementById("mytag");
```

```
var MyResult = document.getElementById("Result");
```

```
MyResult.innerHTML = "color = " + MyTag.style.getPropertyValue("color");
```

```
}
```

```
</script>
```

.....

```
<button onclick="Del();">delete</button>
```

```
<h2 id="mytag" style="background-color:#2062a8; color:#fff; text-align:center;">parsa ghorbanian</h2>
```

```
<script>
```

```
function Del(){
```

```
var MyTag = document.getElementById("mytag");
```

```
MyTag.style.removeProperty("background-color");
```

```
MyTag.style.removeProperty("color");
```

```
MyTag.style.removeProperty("text-align");
```

```
}
```

```
</script>
```

.....