

JavaScript



JAVASCRIPT OOP

Ví dụ 1: Đối tượng trong game

• • •



Danh sách các đối tượng

Nhân vật
Quái
Trụ lửa
Trụ năng lượng
Màn chơi

JAVASCRIPT OOP

Ví dụ 2: Đối tượng Nhân viên 1

• • •



Mô tả một đối tượng nhân viên

Mã nhân viên : ms001

Tên nhân viên : Bưởi

CMND: 0234212

Email: buoi@gmail.com

SĐT : 0993343

Giới tính: Nữ

ĐỐI TƯỢNG (OBJECT)

- Định nghĩa đối tượng:
 - ❖ JavaScript có các kiểu dữ liệu cơ bản: string, number, Boolean, null, undefined...
 - ❖ Tuy nhiên, không có kiểu dữ liệu nào mô tả một đối tượng thực tế.
⇒ **Kiểu dữ liệu Object** ra đời
- Thành phần của đối tượng
 - ❖ Property (thuộc tính): là các đặc điểm của đối tượng (vd: tên nhân viên, lương, màu sắc...)
 - ❖ Method (phương thức): là các hành động của đối tượng (vd: đi, đứng, chạy, tính toán...)

Đối tượng bao gồm gì?

Mã nhân viên
Tên nhân viên
CMND
Email
SĐT
Giới tính

*tinhLuong();
tongGioLam();
tongNgayNghi();*

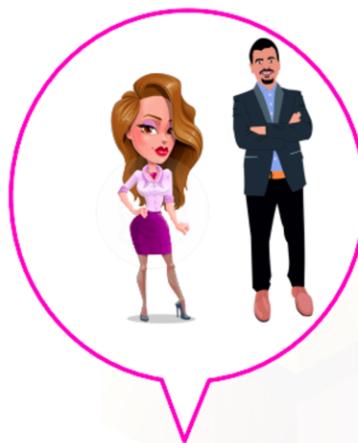
THUỘC TÍNH

PHƯƠNG THỨC

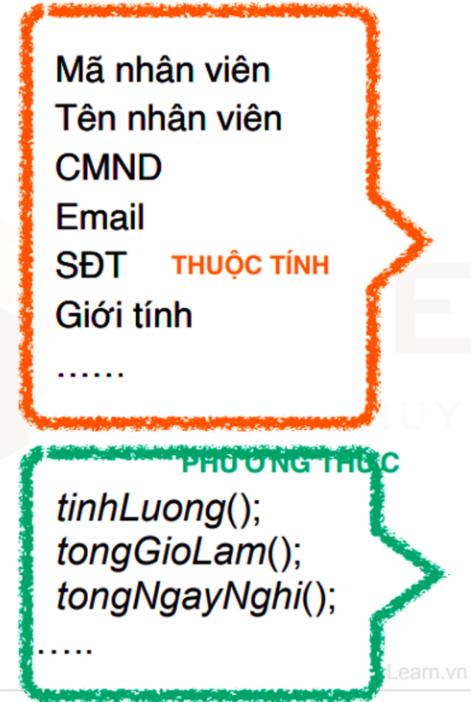


ĐỐI TƯỢNG (OBJECT)

- Các đối tượng trong javascript cũng là các biến chứa nhiều giá trị, mỗi giá trị là 1 cặp **name: value** và cách nhau bằng dấu phẩy ,

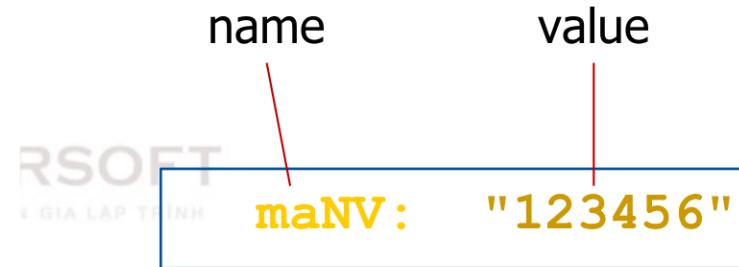


Lớp: Nhân Viên



ĐỐI TƯỢNG (OBJECT)

```
var nhanVien = {  
    //Thuộc tính  
    maNV: "123456",  
    tenNV: "Phuc Nguyen",  
    cmnd: "123456789",  
    email: "dpnguyen53@gmail.com",  
    sdt: "123456789",  
    gioiTinh: "nam",  
  
    //Phương thức  
    tinhLuong: function () {},  
    tongGioLam: function () {},  
    tongNgayNghi: function () {},  
};
```



ĐỐI TƯỢNG (OBJECT)

- Truy cập vào thuộc tính của đối tượng theo một trong hai cách sau:

```
<tên-đối-tượng>.<tên-thuộc-tính>
```

```
<tên-đối-tượng>[ "<tên-thuộc-tính>" ]
```

```
console.log(student.firstName);  
console.log(student["firstName"]);
```

Bài tập đối tượng

- Phân tích đề bài có bao nhiêu đối tượng?
- Gợi ý: Tạo đối tượng sinhVien.
 - Thuộc tính: maSV, tenSV, loaiSV, diemToan, diemVan
 - Phương thức: diemTB(), xepLoai()

Quản Lý Sinh viên

Mã sinh Viên :

Tên sinh Viên :

Loại Sinh Viên :

Điểm Toán:

Điểm Văn:

Hiển Thị Thông Tin

Thông Tin sinh Viên

Tên sinh Viên:

Mã sinh Viên:

Loại SV:

Điểm Trung Bình:

Xếp Loại :

[Link layout bài luyện](#)

LỚP ĐỐI TƯỢNG

- ❑ Chúng ta đã tìm hiểu qua đối tượng (Object). Mở rộng hơn của kiểu Object ta sẽ có khái niệm **Lớp đối tượng**
- ❑ Ví dụ thực tế: Trong lớp học có nhiều Sinh Viên (30 sinh viên). Chẳng lẽ phải tạo ra 30 đối tượng **sinhVien**. Lúc này có một định nghĩa mới đó là **Lớp Đối Tượng** Sinh Viên. Các đối tượng **sinhVien** sẽ được khởi tạo từ **Lớp Đối Tượng** Sinh Viên.
=> **Lớp đối tượng** mô tả cho 1 nhóm các đối tượng có cùng các đặc điểm (thuộc tính) và hành động (phương thức) giống nhau.

JAVASCRIPT OOP



PHƯƠNG THỨC KHỞI TẠO (CONSTRUCTOR)

- Constructor là một method đặc biệt dùng để khởi tạo đối tượng (Object).
- Constructor được gọi tại thời điểm Object được tạo.
- Không có kiểu trả về
- Có thể có tham số hoặc không có tham số.
- Đối với JS cơ bản (ES5), phương thức Constructor không được khai báo một cách rõ ràng như các ngôn ngữ lập trình khác(Java, C, PHP...), bản thân Lớp đối tượng chính là hàm khởi tạo (Constructor).
- Từ phiên bản nâng cấp ES6 trở lên, hàm khởi tạo (Constructor) sẽ được khai báo rõ ràng hơn

LỚP ĐỐI TƯỢNG

- ❑ Ví dụ: tạo lớp đối tượng SinhVien, đồng thời lớp đối tượng SinhVien cũng là hàm khởi tạo (CONSTRUCTOR) có 7 params truyền vào.

```
//Tạo lớp đối tượng sinh viên
function SinhVien(_maSV, _tenSV, _loaiSV, _diemToan, _diemLy, _diemHoa, _diemRenLuyen) {
    //key value
    this.maSV = _maSV;
    this.tenSV = _tenSV;
    this.loaiSV = _loaiSV;
    this.diemToan = _diemToan;
    this.diemLy = _diemLy;
    this.diemHoa = _diemHoa;
    this.diemRenLuyen = _diemRenLuyen;

    this.tinhDTB = function () {};
}
```



Instance - Thể Hiện Lớp ĐỐI TƯỢNG

Từ khóa

```
var sinhVien1 = new SinhVien();
```

```
var sinhVien2 = new SinhVien();
```

```
var sinhVien3 = new SinhVien();
```

INSTANCE - THỂ HIỆN CỦA LỚP ĐỐI TƯỢNG

1. Muốn xài được lớp đối tượng → Phải tạo các **đối tượng cụ thể**. Một trường hợp cụ thể của lớp đối tượng đối tượng → Thể hiện của lớp đối tượng (**Instance** of Class)
2. Sử dụng từ khóa ***new*** để tạo ra thể hiện của lớp đối tượng. Lúc này máy tính sẽ cấp phát một vùng nhớ cho đối tượng

MỘT SỐ GHI NHỚ

- ❑ Đối tượng : **Object**
- ❑ Lớp đối tượng: **function**
- ❑ Thuộc tính/ Dữ liệu/ Biến thành viên: **Attribute/ Data members**
- ❑ Phương thức (**Method**)
- ❑ Thể hiện của lớp đối tượng (**Instance**): Sử dụng từ khóa **new**
- ❑ Naming Convention:
 - **Đối tượng**: Chữ cái đầu viết thường: **nhanVien**, **sinhVien**, **student** (Danh từ)
 - **Lớp đối tượng**: Viết hoa chữ cái đầu: **NhanVien**, **SinhVien**, **Student** (Danh từ)
 - **Thuộc tính**: Chữ cái đầu viết thường
 - **Phương thức**: Chữ cái đầu viết thường

LET'S CODE NOW

- Bước 1 tạo lớp đối tượng SinhVien (không có tham số)

```
function SinhVien() {  
    this.hoTen = "";  
    this.email = "";  
}
```



this → con trỏ của **thể hiện (instance)** của lớp hiện hành, khi nào xài ta gọi **new** và khi đó biến đối tượng sẽ chính là **this**



LET'S CODE NOW

- Vào file main.js code như sau:

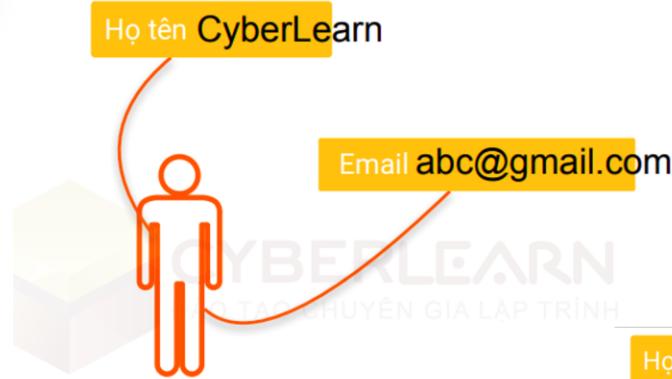
```
var sv1 = new SinhVien();
sv1.hoTen = "CyberLearn";
sv1.email = "abc@gmail.com";

var sv2 = new SinhVien();
sv2.hoTen = "CyberSoft";
sv2.email = "cyber@gmail.com";

console.log("Tên: " + sv1.hoTen + " - Email: " + sv1.email);
console.log("Tên: " + sv2.hoTen + " - Email: " + sv2.email);
```



GIẢI THÍCH CODE



Tên: CyberLearn - Email: abc@gmail.com

Tên: CyberSoft - Email: cyber@gmail.com

Họ tên CyberSoft



Email cyber@gmail.com

LET'S CODE NOW

- Bước 1 tạo lớp đối tượng SinhVien (có tham số)

SinhVien.js > ...

```
1 function SinhVien(_hoTen, _email) {  
2     this.hoTen = _hoTen;  
3     this.email = _email;  
4 }
```



this → con trỏ của **thể hiện (instance)** của lớp hiện hành, khi nào xài ta gọi **new** và khi đó biến đối tượng sẽ chính là **this**



LET'S CODE NOW

- Vào file main.js code như sau:

```
var sv1 = new SinhVien("CyberLearn", "abc@gmail.com");
var sv2 = new SinhVien("CyberSoft", "cyber@gmail.com");

console.log("Tên: " + sv1.hoTen + " - Email: " + sv1.email);
console.log("Tên: " + sv2.hoTen + " - Email: " + sv2.email);
```



DỰ ÁN QUẢN LÝ SINH VIÊN

Quản Lý sinh viên

Mã Sinh Viên :

Tên Sinh Viên :

Email :

Mật khẩu:

Ngày Sinh :

 mm/dd/yyyy CALENDAR

Khóa học:

 Chọn khóa học DOWN ARROW

Điểm toán :

Điểm lý :

Điểm hóa :

Thêm Sinh Viên

Mã Sinh Viên

Tên Sinh Viên

Email

Ngày Sinh

Khóa Học

Điểm Trung Bình

GIẢI THUẬT - DỰ ÁN QUẢN LÝ SINH VIÊN

- Xác định lớp đối tượng
- Vẽ sơ đồ lớp UML
- Liệt kê các thuộc tính
- Liệt kê các phương thức đã thấy trước
- Xây dựng các hàm xử lý: đầu vào, xử lý, đầu ra
- Xây dựng phương thức xử lý nghiệp vụ: tính điểm trung bình, thêm sinh viên, xóa sinh viên, cập nhật sinh viên, tìm kiếm sinh viên.
- Gọi các phương thức xử lý theo yêu cầu bài toán.

Kiểm tra hợp lệ (Validation)

- Kiểm tra dữ liệu bắt buộc nhập
- Kiểm tra dữ liệu là chữ
- Kiểm tra dữ liệu là số
- Kiểm tra email hợp lệ
- Kiểm tra chiều dài chuỗi ký tự

CYBERSOFT
HỌ TẠO CHUYÊN GIA LẬP TRÌNH

Kiểm tra hợp lệ (Validation)

- ❑ Hàm kiểm tra dữ liệu bắt buộc nhập

```
function tên_hàm(){
    //hoặc lấy bằng tag name, class name, querySelector...
    var empt = document.getElementById("text1").value;

    if(empt == ""){
        //Dữ liệu không hợp lệ
        //Thông báo nội dung lỗi
        document.getElementById("text2").innerHTML =
        "Please input a value";
        return false;
    }else{
        //Dữ liệu hợp lệ
        document.getElementById("text2").innerHTML = "";
        return true;
    }
}
```

Regular Expression

Biểu thức chính quy – **Regular Expression**: là một chuỗi ký tự đặc biệt được định nghĩa để tạo nên các mẫu (**pattern**) dùng để phân tích cú pháp, sự trùng khớp, tìm kiếm và thay thế trong các chuỗi, đoạn ký tự.

Regular Expression được dùng trong hầu hết các ngôn ngữ lập trình hiện nay, nó là một công cụ vô cùng mạnh mẽ để phân tích, validate dữ liệu đầu vào.

Regular Expression

Regex flags

- **g** (global): Tìm kiếm trong toàn chuỗi.
- **i** (case insensitive): Không phân biệt hoa thường.
- **m** (multiline): hoạt động trên nhiều dòng.
- Cú pháp cơ bản

Regex trong javascript

• Cách tạo:

Cách 1: `/<pattern>/flags`

Cách 2: `new RegExp(pattern, flags)`

• Sử dụng:

`regexp.exec(string)`: Hàm tìm kiếm chuỗi phù hợp so với regex. Nó trả về một mảng chứa kết quả hoặc null

`regexp.test(string)`: Hàm kiểm tra chuỗi có khớp với regex. Nó trả về true hoặc false

Kiểm tra hợp lệ (Validation)

☐ Hàm kiểm tra dữ liệu chữ

```
function tên_hàm(){
    //hoặc lấy bằng tag name, class name, querySelector...
    var inputVal = document.getElementById("text1").value;
    // Sử dụng lớp đối tượng RegExp của javascript
    var letters = new RegExp("^[A-Za-z]+$");
    if (letters.test(inputVal)) {
        //Dữ liệu hợp lệ
        document.getElementById("text2").innerHTML = "";
        return true;
    } else {
        //Dữ liệu không hợp lệ
        document.getElementById("text2").innerHTML = "Please input alphabet
characters only";
        return false;
    }
}
```

Kiểm tra hợp lệ (Validation)

☐ Kiểm tra dữ liệu là số

```
function tên_hàm(){
    //hoặc lấy bằng tag name, class name, querySelector...
    var inputVal = document.getElementById("text1").value;

    // Sử dụng chuỗi Regular expression
    var numbers = /^[0-9]+$/;
    if (inputVal.match(numbers)) {
        //Dữ liệu hợp lệ
        document.getElementById("text2").innerHTML = "";
        return true;
    } else {
        //Dữ liệu không hợp lệ
        document.getElementById("text2").innerHTML = "Please input numeric
characters only";
        return false;
    }
}
```

Kiểm tra hợp lệ (Validation)

☐ Kiểm tra email hợp lệ

```
function tên_hàm(){
    //hoặc lấy bằng tag name, class name, querySelector...
    var inputVal = document.getElementById("text1").value;

    // Sử dụng chuỗi RegExp
    var mailformat = /^w+([\.-]?\w+)*@\w+([\.-]?\w+)*(\.\w{2,3})+$/;
    if (inputVal.match(mailformat)) {
        //Dữ liệu hợp lệ
        document.getElementById("text2").innerHTML = "";
        return true;
    } else {
        //Dữ liệu không hợp lệ
        document.getElementById("text2").innerHTML = "You have entered an in
valid email address!";
        return false;
    }
}
```

Kiểm tra hợp lệ (Validation)

❑ Kiểm tra chiều dài chuỗi ký tự

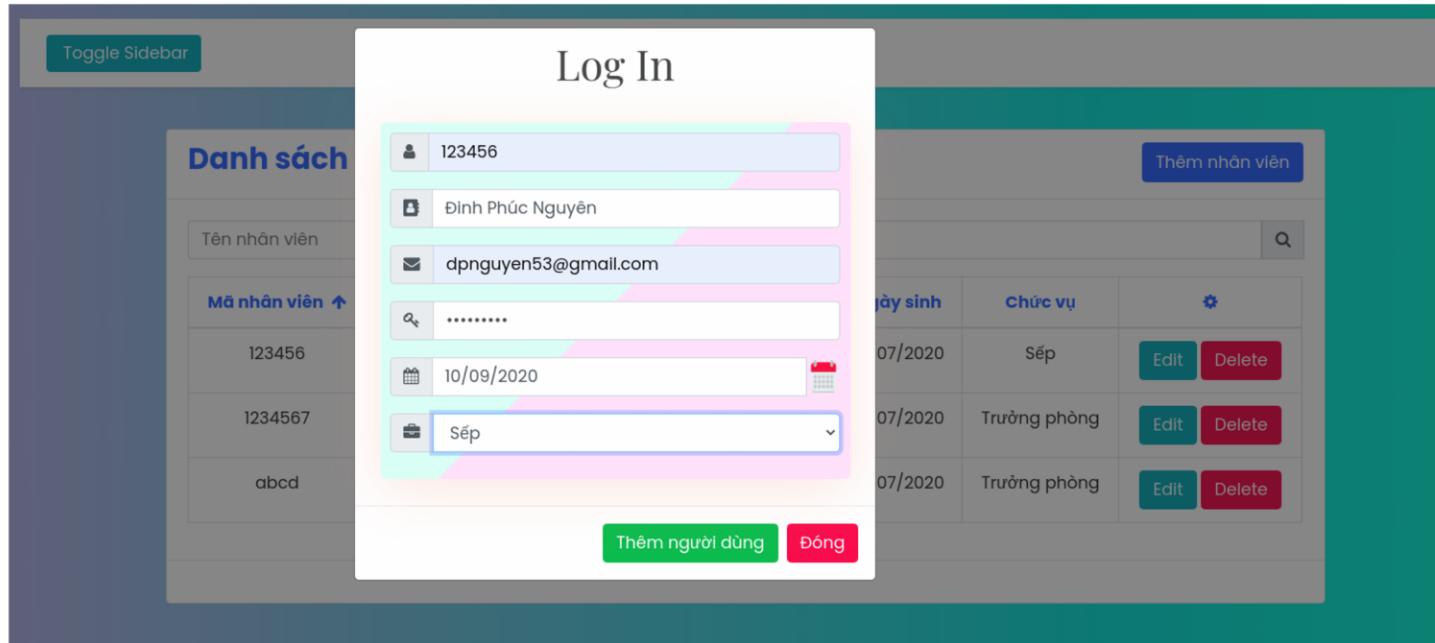
```
function tên hàm(inputtxt, minlength, maxlength)
{
    var field = inputtxt.value;
    var mnlen = minlength;
    var mxlen = maxlength;
    if (field.length > mnlen && field.length < mxlen) {
        //Dữ liệu hợp lệ
        document.getElementById("text2").innerHTML = "";
        return true;
    } else {
        //Dữ liệu không hợp lệ
        document.getElementById("text2").innerHTML = "Please input the
text between " + mnlen + " and " + mxlen + " characters";
        return false;
    }
}
```

Prototype

☐ BỒ SUNG THUỘC TÍNH & PHƯƠNG THỨC

```
function Person(first, last, age, eyecolor) {  
    this.firstName = first;  
    this.lastName = last;  
    this.age = age;  
    this.eyeColor = eyecolor;  
}  
  
Person.prototype.nationality = "English";  
  
function Person(first, last, age, eyecolor) {  
    this.firstName = first;  
    this.lastName = last;  
    this.age = age;  
    this.eyeColor = eyecolor;  
}  
  
Person.prototype.name = function() {  
    return this.firstName + " " + this.lastName;  
};
```

DỰ ÁN QUẢN LÝ NHÂN VIÊN



DỰ ÁN QUẢN LÝ NHÂN VIÊN

Danh sách nhân viên

Thêm nhân viên

Tên nhân viên 🔍

Mã nhân viên ↑	Họ và tên nhân viên	Email	Ngày sinh	Chức vụ	⋮
123456	Nguyen	dpnguyen53@gmail.com	10/07/2020	Sếp	<button>Edit</button> <button>Delete</button>
1234567	Phong Le	dpnguyen53@gmail.com	10/07/2020	Trưởng phòng	<button>Edit</button> <button>Delete</button>
abcd	Dat G	dpnguyen53@gmail.com	10/07/2020	Trưởng phòng	<button>Edit</button> <button>Delete</button>

GIẢI THUẬT - DỰ ÁN QUẢN LÝ NHÂN VIÊN

- Xác định các lớp đối tượng
- Vẽ sơ đồ lớp UML
- Liệt kê các thuộc tính
- Liệt kê các phương thức đã thấy trước
- Xây dựng các hàm xử lý: đầu vào, xử lý, đầu ra
- Xây dựng phương thức xử lý nghiệp vụ: thêm nhân viên, xóa nhân viên, cập nhật nhân viên, tìm kiếm nhân viên theo tên, kiểm tra dữ liệu nhập
- Gọi các phương thức xử lý theo yêu cầu bài toán.