Đối tượng trong JavaScript

Tạo và sử dụng các đối tượng trong JavaScript gần giống với các khái niệm trong lập trình hướng đối tượng, như thuộc tính, phương thức

Các đối tượng - Object

Các biến trong `JavaScript` chứa các dữ liệu, các đối tượng cũng tương tự như vậy, nhưng nó chứa được nhiều giá trị. Hãy nghĩ một đối tượng là một danh sách các giá trị được viết theo cặp `nameobject:value`

Ví dụ sau tạo ra đối tượng lưu trong biến `person` :

```
var person = {
    name:       "John",
    age:        31,
    favColor:   "green",
    height:     183
};
```

Các thuộc tính của đối tượng trên là `name`, `age`, `favColor`, `height` ...

Truy cập đến thuộc tính đối tượng

Có 2 cách để truy cập đến thuộc tính đối tượng để lấy giá trị thuộc tính hoặc gán giá trị vào thuộc tính, ví dụ:

```
objectName.propertyName                 //Cách 1
objectName['propertyName']              //Cách 2
var person = {
    name: "John", age: 31,
    favColor: "green", height: 183
};

var x = person.age;
var y = person['age'];
```

Khởi tạo đối tượng

Cách 1) Khởi tạo cố định

Cách thứ nhất là tạo bằng phương pháp cố định, là cách đã thực hiện ở trên (*tạo đối tường và khởi tạo luôn các thuộc tính cần có*)

```
var person = {
    name: "John",
    age: 42,
```

```
    favColor: "green"
};
```

Cách 2) Khởi tạo bằng hàm tạo

Với cách thứ 2 này bạn khai báo một hàm gọi là hàm tạo rồi tạo ra đối tượng bằng cú pháp `new hamtao()`

Trong hàm tạo hoặc các hàm thuộc đối tượng, sử dụng từ khóa this để tham khảo đến đối tượng, thông qua nó truy cập các thuộc tính *(ý nghĩa của this giống this trong Java, Php, C# ...)*

```
function person(name, age, color) {          // Hàm khởi tạo
  this.name = name;                          // this tham khảo đến đối tượng cần tạo
  this.age = age;
  this.favColor = color;
}

var p1 = new person("John", 42, "green");    // tạo đối tượng
var p2 = new person("Amy", 21, "red");       // tạo đối tượng

document.write(p1.age);                      // Outputs 42
document.write(p2.name);                     // Outputs "Amy"
```

Phương thức trong đối tượng

Một đối tượng ngoài các thuộc tính ra nó còn chứa hàm gọi là phương thức, ví dụ truy cập một hàm

```
objectName.methodName()
```

Như bạn đã biết khi viết chuỗi bạn có sử dụng đến `document.write()` thì `write` chính là phương thức của đối tượng `document`

Phương thức thuộc về một đối tượng, đối tượng này được hàm tham khảo qua từ khóa `this`

Thường các phương thức được định nghĩa qua một hàm khởi tạo đối tượng.

```
//Hàm khởi tạo đối tượng
function person(name, age) {
  this.name = name;
  this.age = age;
  this.changeName = function (name) {
    this.name = name;
  }
}

//Tạo đối tượng
var p = new person("David", 21);

p.changeName("John");
```

```
//Giờ p.name bằng "John"
```

Các phương thức bạn cũng có thể định nghĩa bên ngoài hàm khởi tạo, ví dụ

```
function person(name, age) {
  this.name= name;
  this.age = age;
  this.yearOfBirth = bornYear; //Gán phương thức bên ngoài
}

//Hàm bên ngoài hàm tạo, hàm này được gán vào đối tượng qua hàm tạo ở trên
function bornYear() {
  return 2016 - this.age;
}

var p = new person("A", 22);
document.write(p.yearOfBirth());
// Outputs 1994
```

Setter và Getter

Một thuộc tính của của đối tượng còn thiết lập nó là hàm `setter` hoặc `getter`, nếu là `setter` nó chỉ được gọi qua toán tử gán giá trị cho nó, nếu là `getter` thì chỉ được gọi khi truy cập lấy giá trị thuộc tính.

Hàm setter định nghĩa bằng cách cho thêm `set`, hàm getter định nghĩa bằng cách cho thêm `get`

```
var obj = {
    age: 0,

    set ageInfo(age) {                              //Định nghĩa setter
        console.log('setter - ' + age);
        this.age = age;
    },

    get ageInfo() {                                 //Định nghĩa getter
        console.log('getter');
        return "Thông tin tuổi: " + this.age;
    }
};

obj.ageInfo = 25;           //Gán -> Tự động gọi settter
alert(obj.ageInfo);         //Không phải gán -> Tự động gọi getter
```

Tạo Setter/Getter trong hàm tạo

Trong trường hợp muốn định nghĩa `setter / getter` trong hàm tạo đối tượng thì bạn cần định nghĩa theo nguyên tắc thêm một thuộc tính vào đối tượng đã có với lệnh `Object.defineProperty`

```
//Một đối tượng đã có tên ob, thêm cho nó setter, getter có tên namepro
Object.defineProperty(ob, 'namepro', {
```

```
    set: function(x) {
        //code setter ở đây
       },
    get: function() {
            //code getter ở đây
       }

});
```

Ví dụ trên định nghĩa lại trong hàm tạo:

```
 function person(age) {
    this.age = 0;
    Object.defineProperty(this, 'ageInfo', {
        set : function (age) {
            console.log('setter - ' + age);
            this.age = age;
        },
        get : function () {
            console.log('getter');
            return "Thông tin tuổi: " + this.age;
        }
    });
 }

var obj = new person(0);

obj.ageInfo = 25;
alert(obj.ageInfo);
```