JavaScript

Ba Nguyễn

ES6

ECMA Script 6 hoặc ES6 là tiêu chuẩn mới nhất được sử dụng trong JavaScript. Nhiều cú pháp và từ khóa mới hiện đã được thêm vào bản cập nhật JavaScript ES6.

Các framework JavaScript phổ biến hiện tại như NodeJS, Angular của Google, ReactJS của Facebook sử dụng các tính năng JavaScript ES6 (và mới hơn).

Tìm hiểu thêm về các tiêu chuẩn JavaScript tại đây [wiki/ECMAScript]

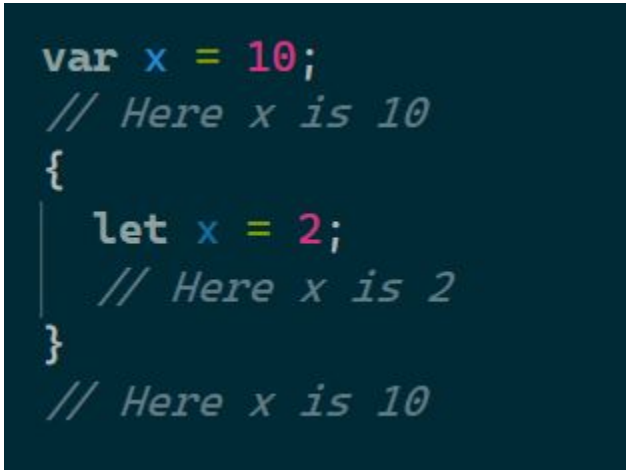
ES6

Các tính năng mới trong JavaScript ES6

- Variable: let, const
- Scope
- Arrow function
- Default parameter
- Rest parameter, spread operator
- Template literal
- Destructuring assignment
- Module
- Class

Variable

Từ khóa let khai báo một biến với block scope, đồng thời biến không thể được khai báo lại



Variable

Từ khóa const khai báo hằng số (còn được gọi là "biến bất biến - immutable"), là các biến không thể gán lại giá trị.

Lưu ý: điều này chỉ làm cho bản thân biến trở nên bất biến chứ không phải giá trị của nó (ví dụ: trong trường hợp giá trị là một đối tượng, điều này có nghĩa là bản thân đối tượng vẫn có thể được thay đổi)

```
const PI = 3.141593;

PI = 3.14; // error

const user = {
    name: "Ba"
};

user.name = "Béo Ú"; // okela
```

Scope

Scope chỉ phạm vi tồn tại của một biến. Một biến chỉ tồn tại bên trong khối (**block**) code mà nó được khai báo. Một **block** là đoạn code được đặt trong cặp dấu **{ }**. **Các cấu trúc điều khiển, hàm, class là một block.**

💡 Riêng biến khai báo với **var** chỉ bị giới hạn trong **block của hàm**

```
{
    let x = 1;
    const y = 2;
    var z = 3;
}
alert(x); // ❌ error
alert(y); // ❌ error
alert(z); // 👌 3
```

```
if (true) {
    let x = 1;
    const y = 2;
    var z = 3;
}
alert(x); // ❌ error
alert(y); // ❌ error
alert(z); // 👌 3
```

```
function a() {
    let x = 1;
    const y = 2;
    var z = 3;
}
alert(x); // ❌ error
alert(y); // ❌ error
alert(z); // ❌ error
```

Scope

Ví dụ với vòng lặp:

```
for (let i = 0; i < 10; i++) {
  // code
}
console.log(i); // ✗ error i is not defined

for (var i = 0; i < 10; i++) {
  // code
}
console.log(i); // i = 10
```

Arrow functions

Arrow function là cú pháp mới, cho phép viết biểu thức hàm ngắn gọn hơn và bao gồm một số tính năng khác biệt so với hàm thông thường

Arrow function không tồn tại từ khóa **this** (hay **arguments**)

```
let secretMessage = () ⇒
    console.log("Hey, I love you 😘😘😘😘");
```

Default Parameters

Chỉ định giá trị mặc định cho tham số

```
function hi(name = "Ba") {
    console.log("I love " + name);
}

hi(); // "I love Ba"
hi("Someone 😙"); // "I love Someone 😙"
```

Rest Parameters

Trong JavaScript, một hàm có thể được gọi với nhiều/ít đối số hơn số lượng tham số được khai báo trong hàm. Cú pháp **rest parameters** cho phép gọi hàm với số lượng đối số bất kỳ, các đối số truyền vào hàm được tổng hợp trong một mảng.

```
let sum = (a, b, ...others) => {
    let total = a + b;

    for (let other of others) {
        total += other;
    }

    return total;
};
```

Spread operator

Phân tách các phần tử của một tập hợp có thể lặp lại - *iterable* (như một mảng hoặc thậm chí một chuỗi) thành cả phần tử/tham số hàm riêng lẻ

```
var params = ["hello", true, 7];
var other = [1, 2, ...params]; // [ 1, 2, "hello", true, 7 ]

function f(x, y, ...a) {
    return (x + y) * a.length;
}
f(1, 2, ...params) === 9;

var str = "foo";
var chars = [...str]; // [ "f", "o", "o" ]
```

Template literals

Template literal (` `) là cú pháp mới, cho phép "nhúng" (nội suy) giá trị của một biến, biểu thức, hoặc thậm chí một phương thức vào chuỗi sử dụng cú pháp ${}

```
var customer = { name: "Foo" };
var card = { amount: 7, product: "Bar", unitprice: 42 };

var message = `Hello ${customer.name},
want to buy ${card.amount} ${card.product} for
a total of ${card.amount * card.unitprice} bucks?`;
```

Destructuring assignment

Destructuring assignment là một cú pháp đặc biệt cho phép “giải nén” các mảng hoặc đối tượng thành một loạt các biến

```
let arr = ["John", "Smith"];

let [firstName, surname] = arr;
```

Destructuring assignment

Destructuring assignment là một cú pháp đặc biệt cho phép “giải nén” các mảng hoặc đối tượng thành một loạt các biến

```
let options = {
    title: "Menu",
    width: 100,
    height: 200,
};

let { title, width, height } = options;

alert(title); // Menu
alert(width); // 100
alert(height); // 200
```