Internet và Công nghệ Web

Chương 4
Ngôn Ngữ Javascript

Nội dung

1. Giới thiệu JS
2. Khai báo biến – In giá trị
3. Kiểm tra kiểu dữ liệu
4. Các kiểu dữ liệu
5. Các toán tử
6. Cấu trúc rẽ nhánh
7. Cấu trúc lặp
8. Hàm
9. Áp dụng JS vào lập trình web
10. DOM - Document Object Model
11. Bài tập

Giới thiệu

- JavaScript là ngôn ngữ thông dịch lập trình phía client (client-side programming language).
- Sử dụng rộng rãi trong phát triển web để tạo tính năng động và tương tác cho trang web.
- Các chức năng chính:
  - Tương tác với trang web: thay đổi nội dung, cấu trúc và kiểu dáng.
  - Xử lý sự kiện: xử lý sự kiện như nhấp chuột, nhập dữ liệu từ bàn phím, gửi dữ liệu lên máy chủ.
  - Làm việc với trình duyệt: tương tác với trình duyệt để thay đổi URL, lưu trữ dữ liệu trong cơ sở dữ liệu cục bộ, thực hiện AJAX để tải dữ liệu từ máy chủ mà không cần tải lại trang.

Comment

- Single Line Comment: ***//***
- Multi-line Comment: ***/* */***

Khai báo biến

- JavaScript tự động quản lý kiểu dữ liệu => không cần phải chỉ định kiểu dữ liệu khi khai báo biến.
- Dùng từ khóa **var**, **let**, hoặc **const** để khai báo.

ES6 (ECMAScript 6) là một phiên bản tiêu chuẩn của JavaScript, cũng được gọi là ECMAScript 2015

- **var**: không còn được khuyến nghị sử dụng trong **ES6** và hiện tại.
- **let**: khai báo biến có phạm vi trong một block khối lệnh hoặc hàm.
- **const**: khai báo một hằng số và không thể thay đổi giá trị.

Khai báo biến

Quy tắc

- Phải bắt đầu bằng một chữ cái hoặc dấu gạch dưới _
- Có thể chứa số, chữ số (A-Z, a-z) và dấu gạch dưới _
- Chỉ cho phép hai ký tự đặc biệt trong tên biến: $ và _

- Không chứa bất kỳ ký tự đặc biệt nào như $, #, @, %, ^,
- Không bắt đầu bằng số.
- Không trùng các từ khóa như let, class, true…

Khai báo biến

Ví dụ

```
var 1abc; // invalid
var _1abc; // valid
var _1_abc; // valid
var _1_abc_; // valid
var $ = 5; // valid
var _$ = 5; // valid
var _1_abc$ = 5; // valid
var _ = "hello"; // valid
var cat123$ = "Some cats"; // valid
var cat123_ = "Some cats"; // valid
var dog@12 = "Some Dog"; // invalid
var abc = 10; // valid
var Abc = 11; // valid
// abc and Abc are different
```

console.log()

- Là một hàm trong JS để hiển thị thông tin hoặc giá trị của biến trong cửa sổ console của trình duyệt.

- Cú pháp

```
console.log(item1, item2, ……, itemN);
```

- Trong đó: item1, item2, ..., itemN là các giá trị hoặc biến muốn hiển thị.
- Ví vụ:

```
let x = 10;
let y = 20;

console.log("Giá trị của x là:", x, "\nGiá trị của y là:", y);
```

Kiểm tra kiểu dữ liệu

Phương pháp

- Sử dụng toán tử typeof, hoặc instanceof để kiểm tra kiểu dữ liệu.

```
let x = 10;
console.log(typeof x); // "number"

let y = 'Hello';
console.log(typeof y); // "string"

let z = true;
console.log(typeof z); // "boolean"

let t;
console.log(typeof t); // "undefined"

let w = null;
console.log(typeof w); // "object"
```

```
let arr = [];
console.log(arr instanceof Array); // true

let obj = {};
console.log(obj instanceof Object); // true


function Person(name) {
    this.name = name;
}

let person = new Person('Nguyen Van A');
console.log(person instanceof Person); // true
```

Kiểm tra kiểu dữ liệu

Phương pháp

- Sử dụng Object.prototype.toString.call(…)

```
let a = [6, 7, 8, 9];
console.log(Object.prototype.toString.call(a)); // [object Array]

let d = { key: 'value' };
console.log(Object.prototype.toString.call(d)); // [object Object]
```

Kiểu dữ liệu

Các loại

- Number: Kiểu dữ liệu số, biểu diễn số nguyên và số thực. Ví dụ: 3, 3.14.
- String: Kiểu dữ liệu chuỗi. Ví dụ: "Công nghệ Thông tin".
- Boolean: Kiểu dữ liệu logic, có 2 giá trị true hoặc false.
- Undefined: Biến chưa được gán giá trị sẽ có kiểu dữ liệu undefined.
- Null: Đại diện cho giá trị rỗng hoặc không tồn tại.
- Object: Kiểu dữ liệu đối tượng, tập hợp các thuộc tính và phương thức. Đối tượng khởi tạo bằng dấu **{ }**, hoặc bằng constructor function.
- Array: Là một kiểu dữ liệu chứa nhiều giá trị có thứ tự. Mảng được khởi tạo bằng dấu ngoặc vuông **[ ]**.
- Date: Kiểu dữ liệu Date cho phép biểu diễn thời gian và ngày tháng.

Kiểu dữ liệu

Các loại

- Symbol (ES6): Kiểu dữ liệu symbol là một giá trị duy nhất, thường được đặt tên cho các thuộc tính đối tượng.
- Map (ES6): cấu trúc dữ liệu để lưu trữ các cặp khóa-giá trị.
- Set (ES6): tập hợp các giá trị duy nhất.
- BigInt (ES11): Kiểu dữ liệu BigInt biểu diễn các số nguyên lớn hơn giới hạn của kiểu số thông thường.
- RegExp: Kiểu dữ liệu biểu thức chính quy (regular expression) để so khớp và xử lý chuỗi.
- Array Buffer (ES6): Kiểu dữ liệu nhị phân.
- Typed Array (ES6): Mảng có kiểu dữ liệu nhị phân.

Khai báo kiểu số

```
let tuoi = 30;
let donGia = 26.33;
let luongCB = 7.6e6;

let x = 5;
let y = 3;
let tong = x + y;
let hieu = x - y;
let tich = x * y;
let thuong = x / y;
let du = x % y;

const PI = 3.14;
const pi = Math.PI;
```

Kiểu số

Các hàm chuyển kiểu

- parseInt(): Chuyển đổi một chuỗi thành kiểu số nguyên (integer).
- parseFloat(): Chuyển đổi một chuỗi thành kiểu số thực (floating-point).
- Number(): Chuyển đổi một giá trị thành kiểu số, có thể chuyển đổi các kiểu dữ liệu khác nhau thành kiểu số.
- parseInt() với cơ số để chuyển đổi chuỗi thành số nguyên với một cơ số.
- Khi chuyển đổi từ chuỗi sang kiểu số, nếu chuỗi không chứa giá trị số hợp lệ thì kết quả là **NaN** (Not-a-Number).

Các hàm chuyển kiểu

```
let strInt = "6789";
let num1 = parseInt(strInt);

let strPI = "3.14";
let num2 = parseFloat(strPI);

let num3 = Number(strInt);

let bool = true;
let num4 = Number(bool);

let strBinary = "1010";
let num5 = parseInt(strBinary, 2);
```

Kiểu số

Các hàm cơ bản

- Math.sqrt()
- Math.pow() //**
- Math.round() để làm tròn số
- Math.floor() làm tròn lên
- Math.ceil() làm tròn xuống

```
let number = 45.6789;

let rounded = Math.round(number); //46
let roundedDown = Math.floor(number); //45
let roundedUp = Math.ceil(number); //46

let precision2 = number.toPrecision(3); // 45.7
let fixed2 = number.toFixed(2); // 4.68
```

- toPrecision(num, n): làm tròn số, n đại diện cho số lượng
- toFixed(num, n) làm tròn số, n là số lượng chữ số sau dấu chấm.

Kiểu chuỗi - String

Khai báo

- Dùng "..." hoặc '...' để khởi tạo chuỗi.

```
let lang = "JavaScript";
let lang2 = 'JavaScript';

let strTB = `Ngôn ngữ ${lang} rất dễ học`;

console.log(strTB);
```

Các hàm xử lý chuỗi

```
let s = "JavaScript rất dễ học";
let s2 = 'JavaScript rất dễ học';

let stringLength = s.length; //length() để lấy độ dài của chuỗi

//indexOf() tìm vị trí của một chuỗi con trong chuỗi gốc
let indexOf1 = s.indexOf("rất"); //11
let indexOf2 = s.indexOf("java"); //-1

//slice() để cắt chuỗi từ vị trí này đến vị trí khác
let slicedString = s.slice(0, 10); // "JavaScript"
```

Các hàm xử lý chuỗi

```
//substring() để cắt chuỗi con dựa trên chỉ số
let substring = s.substring(11, 17); // "rất dễ"

//replace() để thay thế chuỗi con trong chuỗi gốc
let replacedString = s.replace("học", "code");

//toUpperCase(), toLowerCase() chuyển đổi chữ hoa và chữ thường
let upperCaseString = s.toUpperCase();
let lowerCaseString = s.toLowerCase();
```

Kiểu Boolean

```
let isTrue = true; // Khai báo biến kiểu boolean với giá trị true
let isFalse = false; // Khai báo biến kiểu boolean với giá trị false

let num = 0;
let str = "Hello";

// Sử dụng toán tử ! (NOT) để kiểm tra giá trị tương đương kiểu boolean
let isNum = !!num; // Kiểm tra giá trị tương đương kiểu boolean của number
let isStr = !!str; // Kiểm tra giá trị tương đương kiểu boolean của str

console.log("isTrue:", isTrue);
console.log("isFalse:", isFalse);
console.log("isNum:", isNum); // false (0 được coi là giá trị "false")
console.log("isStr:", isStr); // true (chuỗi không rỗng được coi là giá trị "true")
```

undefined

- undefined đại diện cho biến đã được khai báo nhưng chưa được gán giá trị.
- Hoặc, undefined là giá trị trả về từ một hàm không có return.

- Ví dụ

```
let tuoi; // Khai báo biến nhưng không gán giá trị
console.log(tuoi); // undefined

let namSinh = undefined; // Sử dụng biến được gán giá trị undefined
console.log(namSinh); // undefined

// Hàm không có lệnh return
function doSomething() {
     // Không có lệnh return
}

console.log(doSomething()); // undefined
```

null

- Kiểu dữ liệu **null** biểu diễn một giá trị không tồn tại hoặc không có giá trị.
- Ví dụ

```
// Khai báo biến có giá trị null
let myNullVariable = null;


// Kiểm tra nếu biến có giá trị null
if (myNullVariable === null) {
    console.log('Biến có giá trị null.');
}
else {
    console.log('Biến không có giá trị null.');
}
```

Array

- Mảng là một cấu trúc dữ liệu để lưu trữ nhiều giá trị (nhiều phần tử).
- Các phần tử trong mảng được truy cập thông qua chỉ số.
- Các phần tử có thể chứa các giá trị cùng hoặc khác kiểu dữ liệu, như chuỗi, số nguyên, số thực, đối tượng, mảng khác, hoặc hàm.

```
// Khai báo một mảng chứa các số nguyên
let num = [1, 2, 3, 4, 5, 6];

// Khai báo một mảng chứa chuỗi
let lang = ['Java', 'Python', 'JavaScript', 'C++'];

// Mảng chứa các kiểu dữ liệu khác nhau
let arr = [10, 'AFC', true, { city: 'Thủ Đức' }, [1, 2, 3]];
```

Array

```
let arr = [1, 2, 3, 4, 5, 6];

console.log(arr[2], "\t", arr[5])

console.log(arr[10]) //undefined

console.log("Các phần tử trong mảng:");
for (let i = 0; i < arr.length; i++) {
    console.log(arr[i]);
}

for (let item of arr) {
    console.log(item);
}
```

Array

```
// Tính tổng của các phần tử trong mảng
let sum = 0;
for (let number of arr) {
    sum += number;
}
console.log("Tổng các số trong mảng:", sum);

// Thêm một phần tử mới vào mảng
arr.push(7);
console.log("Mảng sau khi thêm phần tử mới:", arr);

// Xóa phần tử cuối cùng khỏi mảng
let item = arr.pop();
console.log("Phần tử bị xóa:", item);
console.log("Mảng sau khi xóa phần tử cuối cùng:", arr);
```

Các phương thức

```
let arr = ['Java', 'Python', 'JavaScript', 'C++'];

// In mảng
console.log("Mảng ban đầu:", arr);

// Phương thức push: Thêm phần tử vào cuối mảng
arr.push('C#');
console.log("Mảng sau thêm:", arr);

// Phương thức pop: Xóa phần tử cuối cùng khỏi mảng
let item = arr.pop();
console.log("Phần tử bị xóa:", item);
console.log("Mảng sau khi xóa:", arr);
```

Các phương thức

```
// Phương thức unshift: Thêm phần tử vào đầu mảng
arr.unshift('R');
console.log("Mảng sau khi thêm:", arr);

// Phương thức shift: Xóa phần tử đầu tiên khỏi mảng
let head = arr.shift();
console.log("Phần tử bị xóa:", head);
console.log("Mảng sau khi xóa:", arr);

// Phương thức slice: Lấy một phần trong mảng
let arr2 = arr.slice(1, 3);
console.log("Mảng mới:", arr2);
```

Xóa, thêm hoặc thay đổi phần tử

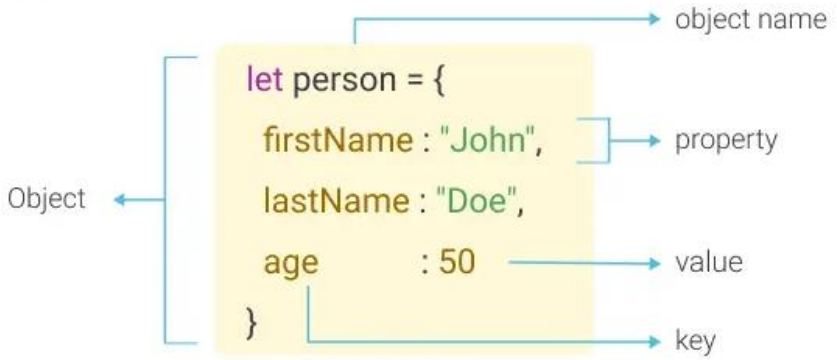
```
array.splice(start, deleteCount, item1, item2, ...);
```

- start: Vị trí bắt đầu cắt hoặc chèn phần tử trong mảng.
- deleteCount: Số lượng phần tử sẽ bị xóa, bắt đầu từ vị trí start. Nếu không xác định (hoặc thiết lập là 0) thì không có phần tử nào bị xóa.
- item1, item2, ...: Các phần tử mới được chèn vào mảng từ vị trí start.

```
let a = ['Java', 'Python', 'JavaScript', 'C++'];
console.log(a);

// Phương thức splice: Xóa, thêm hoặc thay đổi phần tử trong mảng
a.splice(1, 2, 'Ruby', 'R');
console.log(a);
```

object

- Kiểu dữ liệu object là một kiểu dữ liệu chính trong JS.
- Kiểu object có 2 thành phần:
  1. Thuộc tính - property
  2. Phương thức - method



- Được xây dựng từ một tập hợp các cặp "key-value" (khóa - giá trị).
  - **key** là một chuỗi.
  - **value** có thể là bất kỳ kiểu dữ liệu nào: chuỗi, số, mảng, đối tượng khác, hoặc là một hàm.

object

Ví dụ

```
let car = {
    brand: 'Suzuki',
    model: 'Wagon',
    year: 2023,
    features: ['GPS', 'Bluetooth'],
    printf: function () {
        console.log(`Model: ${this.model}`);
    }
};

console.log(car.brand);
console.log(car.features[1]);
car.printf();
```

object

Ví dụ

```
let person = {
    name: 'Nguyễn Văn A',
    age: 39,
    email: 'nguyen@abc.com',
    address: {
        street: '729 Vành Đai',
        city: 'Hồ Chí Minh',
        country: 'Việt Nam'
    },
    interests: ['Xem phim', 'Nghe nhạc', 'Nấu ăn'],

    // Tạo phương thức trong đối tượng
    greet: function () {
        console.log(`Tên ${this.name} \nTuổi ${this.age}`);
    }
};
```

```
console.log(person.name);
console.log(person.age);
console.log(person.address.city);
console.log(person.interests[0]);

// Gọi phương thức
person.greet();
```

Date

Giới thiệu

- Kiểu dữ liệu Date là một kiểu dữ liệu được sử dụng để biểu diễn thời gian và ngày tháng.
- Thực hiện các thao tác liên quan đến ngày, tháng, năm, giờ, phút, giây và miligiây.

```
const dt = new Date(); // Lấy thời gian hiện tại

const dt2 = new Date(2023, 10, 3); // Tạo một thời điểm cụ thể

// Lấy thông tin về ngày, tháng, năm
const year = dt.getFullYear();
const month = dt.getMonth(); // Lưu ý: Tháng bắt đầu từ 0
const day = dt.getDate();

console.log(`Ngày ${day} Tháng ${month + 1} Năm ${year}`);
```

Trừ 2 ngày với nhau

```
const dtNow = new Date();

console.log("Ngày hiện tại:", dtNow);


const dt = new Date(2023, 8 - 1, 20);
console.log("Ngày cụ thể:", dt);

// Khoảng cách giữa hai ngày
const Time = dt.getTime() - dtNow.getTime();
const Days = Time / (1000 * 3600 * 24); // Chuyển từ mili giây
sang ngày
console.log("Khoảng cách giữa hai ngày là:", Days, "ngày.");
```

Ngày hiện tại: Fri Nov 03 2023 23:54:17 GMT+0700 (Indochina Time)
Ngày cụ thể: Sun Aug 20 2023 00:00:00 GMT+0700 (Indochina Time)
Khoảng cách giữa hai ngày là: -75.9960384837963 ngày.

Các toán tử

Toán tử số học

+	Cộng
-	Trừ
*	Nhân
/	Chia
%	Lấy phần dư
**	Lũy thừa

Các toán tử

Toán tử gán

=	Gán giá trị
+=	Cộng rồi gán lại
-=	Trừ rồi gán lại
*=	Nhân rồi gán lại
/=	Chia rồi gán lại
%=	Lấy phần dư rồi gán lại

Các toán tử

Toán tử so sánh

==	So sánh giá trị (không kiểm tra kiểu dữ liệu)
===	So sánh giá trị và kiểu dữ liệu (kiểm tra cả giá trị và kiểu dữ liệu)
!=	Không bằng (so sánh giá trị, không kiểm tra kiểu dữ liệu)
!==	Không bằng (so sánh giá trị và kiểu dữ liệu)
>	Lớn hơn
<	Nhỏ hơn
>=	Lớn hơn hoặc bằng
<=	Nhỏ hơn hoặc bằng

Các toán tử

Toán tử logic

&&	AND logic (Và)
||	OR logic (Hoặc)
!	NOT logic (Phủ định)

Các toán tử

Toán tử ba ngôi (ternary operator)

- Cú pháp

condition ? expression1 : expression2

- Ví dụ:

```
let soSinhVien = 55;

let soThanhVien = soSinhVien <= 50 ? 3 : 5;

console.log("Số thành viên trong nhóm:", soThanhVien);
```

Cấu trúc rẽ nhánh

if một nhánh

```
let nongDoCon = 0;
let bangLai = true;

if (nongDoCon <= 0 && bangLai == true) {
    console.log("Được phép lái xe.");
}
```

```
let nongDoCon = 0;
let bangLai = true;

if (nongDoCon <= 0) {
    if (bangLai == true) {
        console.log("Được phép lái xe.");
    }
}
```

Cấu trúc rẽ nhánh

if hai nhánh

```
let nongDoCon = 0;
let bangLai = true;

if (nongDoCon <= 0 && bangLai == true) {
    console.log("Được phép lái xe.");
}
else {
    console.log("KHÔNG được phép lái xe.");
}
```

Cấu trúc rẽ nhánh

Cú pháp if nhiều nhánh

```
if (điều_kiện_1) {

}
else if (điều_kiện_2) {

}
else if (nđiều_kiện_3) {

}
else if (điều_kiện_n) {

} else {

}
```

Cấu trúc rẽ nhánh

Cấu trúc switch

```
let day = 6;

switch (day) {
    case 2:
        console.log("Thứ hai");
        break;
    case 4:
        console.log("Thứ tư");
        break;
    case 6:
        console.log("Thứ sáu");
        break;
    default:
        console.log("Không phải thứ 2, 4, 6");
}
```

Cấu trúc lặp

Cú pháp for

```
for (start; stop; step) {
    // Lệnh thực thi trong vòng lặp
}
```

Ví dụ

```
for (let i = 1; i <= 5; i++) {
    console.log(i);
}
```

Cú pháp while

```
while (condition) {
    // Lệnh thực thi trong vòng lặp
}
```

- Ví dụ

```
let i = 1;
while (i <= 5) {
    console.log(i);
    i++;
}
```

Cú pháp do … … while

```
do {
    // Lệnh thực thi ít nhất một lần
} while (condition);
```

- Ví dụ

```
let i = 1;

do {
    console.log(`Giá trị của i là: ${i}`);
    i++;
} while (i <= 6);
```

Cấu trúc lặp

Cú pháp for … … in

- Vòng lặp for...in để lặp qua các thuộc tính của một đối tượng.

```
for (variable in object) {
    // Lệnh thực thi trong vòng lặp
}
```

- Ví dụ

```
let person = {
    firstName: "Nguyễn",
    lastName: "Văn A",
    age: 39,
    job: "Chuyên viên"
};

for (let key in person) {
    console.log(`${key}: ${person[key]}`);
}
```

Cấu trúc lặp

Cú pháp for … … of

- Vòng lặp for...of để lặp qua các phần tử của một mảng hoặc các giá trị có thể lặp như chuỗi, mảng, Map, Set…

```
for (variable of iterable) {
    // Lệnh thực thi trong vòng lặp
}
```

- Ví dụ

```
let number = [1, 2, 3, 4, 5, 6];

for (let num of number) {
    console.log(num);
}
```

Function

Cú pháp

```
function functionName(parameter1, parameter2...) {
    // Đoạn code thực thi của hàm
    // Ví dụ:
    var result = parameter1 + parameter2;
    return result;
}
```

- functionName là tên của hàm.
- parameter1, parameter2… là các tham số thể hiện đầu vào, có thể không có hoặc nhiều hơn.
- return …: trả về kết quả của hàm, có thể trả về bất kỳ giá trị hoặc không có return.

Ví dụ

```
function tinh(a, b) {
    var tong = a + b;
    return tong;
}

// Gọi hàm và lấy kết quả
var ketQua = tinh(6, 9);

console.log(ketQua);
```

Cú pháp theo ES6 (ES2015)

- Sử dụng cú pháp "arrow function" (hàm mũi tên).

```
const functionName = (parameter1, parameter2...) => {
    // Đoạn code thực thi của hàm

    let result = parameter1 + parameter2;
    return result;
};
```

Ví dụ

```
const tinh = (a, b) => {
    let tong = a + b;
    return tong;
};

// Gọi hàm và lấy kết quả
let ketQua = tinh(7, 9);
console.log(ketQua);
```

Function

Ví dụ

```
const myFunction = () => {
    console.log("Xin chào ES6");
};

const greet = name => {
    console.log(`Xin chào, ${name}.`);
};

const cong = (a, b) => {
    return a + b;
};

const nhan = (a, b) => a * b;
```

```
// Gọi các hàm
myFunction();
greet("ES6");
console.log(cong(5, 3));
console.log(nhan(4, 6));
```

Áp dụng JS vào lập trình web

Vị trí

Trực tiếp trong HTML

- JavaScript có thể được nhúng trực tiếp trong HTML bằng cách sử dụng thẻ <script> trong phần <head> hoặc <body> của trang web.

Tập tin js

- JavaScript được viết trong các tệp .js riêng biệt và sau đó được liên kết vào tài liệu HTML.

Output

- element.innerHTML
- document.write()
- alert(): Hộp thoại
- console.log()

In - Out

```
<script language="javascript">
    var x = "John";
    alert("Hello World!. Dùng alert" + "ededec" + x);
    //Xuất hộp thoại thông báo. Trong đó nút OK
</script>

<script language="javascript">
    confirm("Do you like ...?. Dùng confirm");
    //Hộp thoại thông báo. Trong đó nút OK, Cancel
</script>

<script language="javascript">
    var t = prompt("Nhập tên của bạn", ''); //Hộp thoại nhập thông tin.
    alert("Chào bạn " + t);
</script>
```

In - Out

```
<script language="javascript">
    var website = 'sonpt.name.vn';
    document.write(website, "Viết bằng write <br> Hihi <br)"); //Viết ra html
    document.write('Hân hành chào các em<br>');

</script>

<script language="javascript">
    console.log('Giá trị: ' + 5) //Dùng để debug
</script>
```

```
<p id="demo"> Kết quả: </p>

<script>
    document.getElementById("demo").innerHTML = "Kết quả: 3";
</script>
```

Xem thêm demo Output

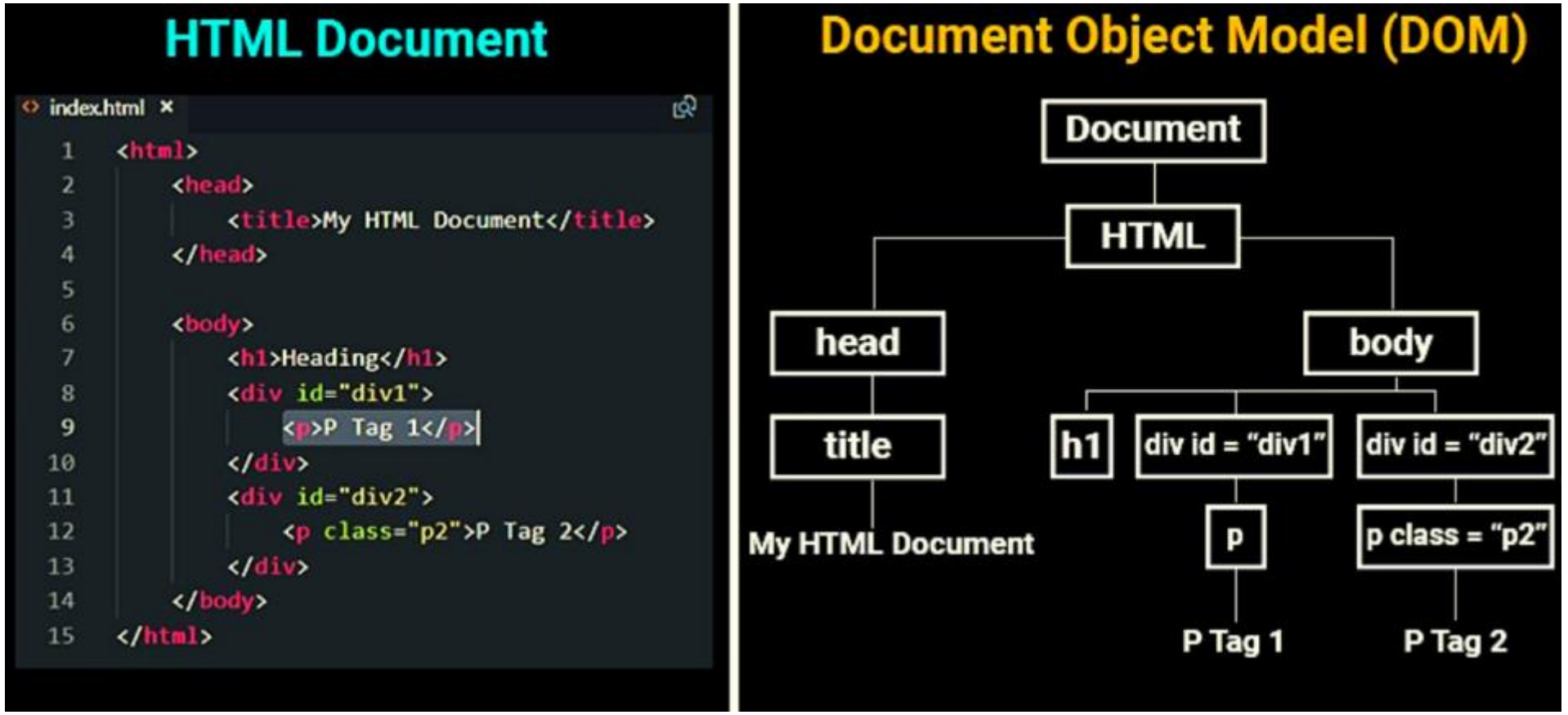
DOM - Document Object Model

Giới thiệu

- DOM là một giao diện lập trình ứng dụng (API) cho phép JavaScript tương tác và thay đổi cấu trúc, nội dung, kiểu dáng của trang web hoặc tài liệu HTML/XML.
- DOM đại diện cho cấu trúc cây của tài liệu web
- Biểu diễn mọi phần tử trong trang web (thẻ HTML, văn bản, hình ảnh…) có thể được truy cập và thay đổi => bằng cách sử dụng các phương thức và thuộc tính.
- DOM tương tác với trình duyệt thực hiện sự kiện như nhấn nút, chuột, hoặc gửi dữ liệu.

DOM - Document Object Model

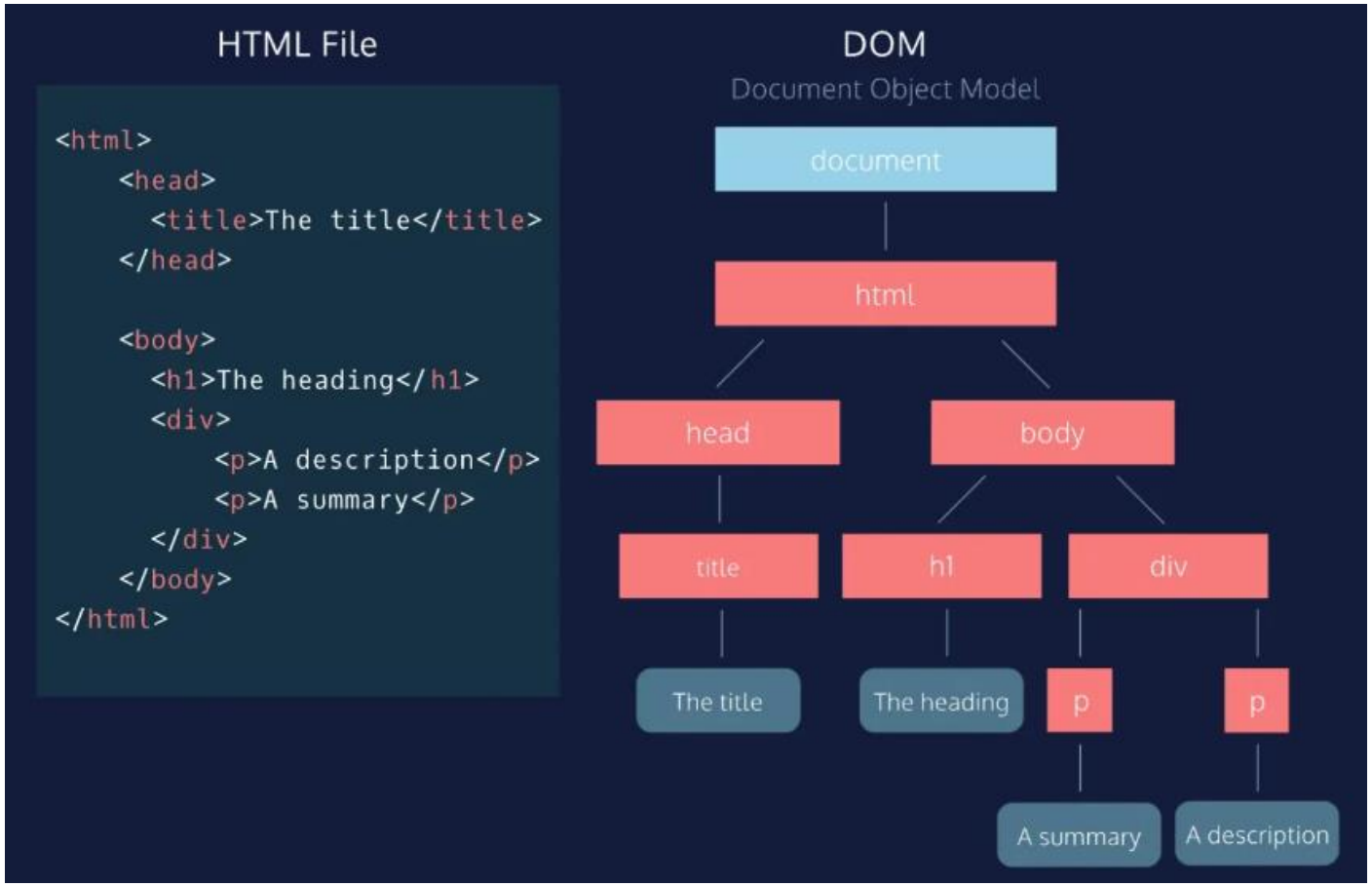
Giới thiệu



Nguồn hình: Internet

DOM - Document Object Model

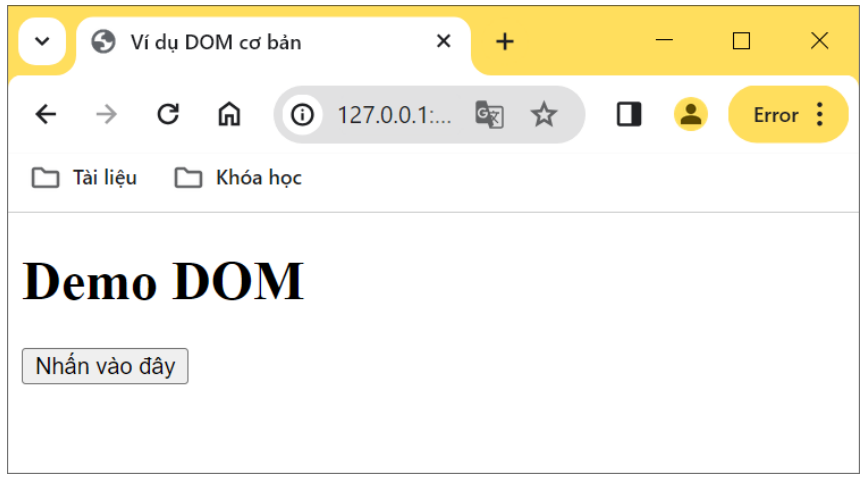
Giới thiệu

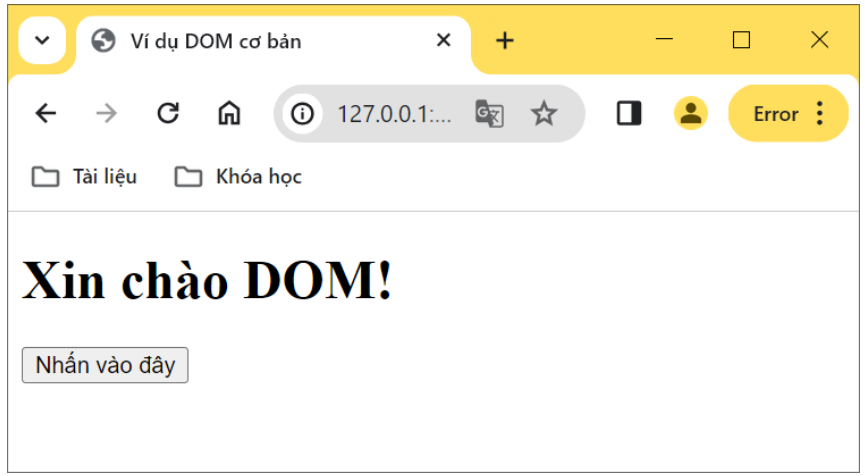


Nguồn hình: Internet

Ví dụ minh họa

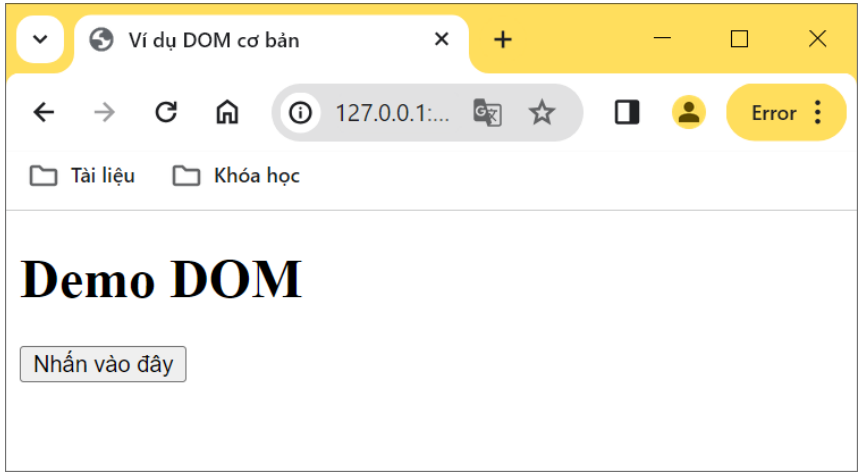
```
<!DOCTYPE html>
<html>
<head>
    <title>Ví dụ DOM cơ bản</title>
</head>
<body>
    <h1 id="myHeading">Demo DOM</h1>
    <button id="myButton">Nhấn vào đây</button>

    <script>
        let heading = document.getElementById("myHeading");
        let button = document.getElementById("myButton");

        button.addEventListener("click", function () {
            heading.innerHTML = "Xin chào DOM!";
        });
    </script>
</body>
</html>
```



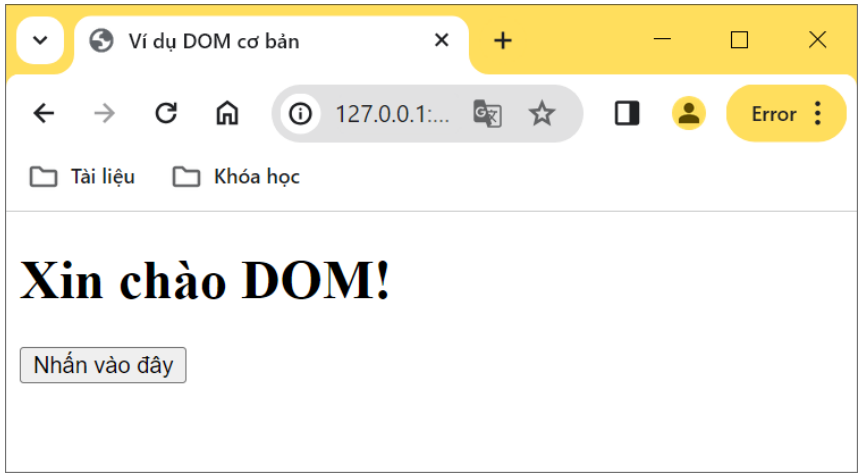


Ví dụ minh họa

```
let heading = document.getElementById("myHeading");

let button = document.getElementById("myButton");

button.addEventListener("click", function () {
    heading.innerHTML = "Xin chào DOM!";
});
```




```
<!DOCTYPE html>
<html>
    <head>
        <title>Ví dụ DOM cơ bản</title>
    </head>
    <body>
        <h1 id="myHeading">Demo DOM</h1>
        <button id="myButton">Nhấn vào đây</button>

        <script src="jscript1.js"></script>
    </body>
</html>
```



Document trong Javascript

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>

<body>
    <p>looping through all the document properties and displaying their values</p>

    <script>
        for (var props in document) {
            console.log(props + ' => ' + document[props]);
        }
    </script>
</body>

</html>
```

Lấy tham chiếu đến phần tử

Method	Description
document.getElementById()	Chọn phần tử duy nhất với **ID** đã cho. Trường hợp có 2 **ID** giống nhau, sẽ chọn phần tử đầu tiên.
document.getElementsByClassName()	Chọn các phần tử theo tên class
document.getElementsByTagName()	Chọn các phần tử theo tên tag
document.querySelector()	Chọn phần tử đầu tiên dựa trên chuỗi CSS.
document.querySelectorALL()	Chọn tất cả các phần tử dựa trên chuỗi CSS.

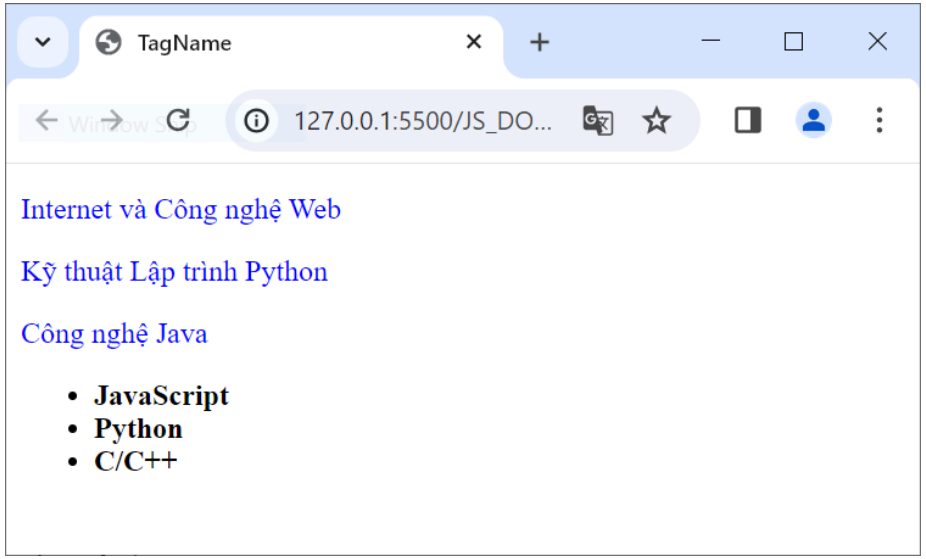
Lấy tham chiếu đến phần tử

Ví dụ

```
<body>
    <p id="id1">element có id = "id1".</p>
    <button onclick="findElement()">Access element</button>

    <script>
        function findElement() {
            const element = document.getElementById("id1");
            element.innerHTML = "Element tìm thấy";
        }
    </script>
</body>
```

```html
<!DOCTYPE html>
<html>
<head>
    <title>ClassName</title>
</head>
<body>
    <h1 class="title">Ngôn ngữ lập trình</h1>
    <p class="content">JavaScript</p>
    <p class="content">Python</p>
    <p class="content">C++</p>
    <button class="btn">Click vào đây</button>

    <script>
        // Lấy tất cả các phần tử có class = "content"
        var contentElements = document.getElementsByClassName("content");

        for (var i = 0; i < contentElements.length; i++) {
            contentElements[i].innerHTML = "Đã cập nhật " + (i + 1);
        }

        // Lấy phần tử có class "btn" và thêm sự kiện click phần tử
        var buttonElement = document.getElementsByClassName("btn")[0];
        buttonElement.addEventListener("click", function () {
            alert("Button đã được gọi.");
        });
    </script>
</body>
</html>
```

```html
<body>
    <p>Internet và Công nghệ Web</p>
    <p>Kỹ thuật Lập trình Python</p>
    <div>
        <p>Công nghệ Java</p>
    </div>
    <ul>
        <li>JavaScript</li>
        <li>Python</li>
        <li>C/C++</li>
    </ul>

    <script>
        var paragraphs = document.getElementsByTagName("p");
        for (var i = 0; i < paragraphs.length; i++) {
            paragraphs[i].style.color = "blue";
        }

        // Lấy tất cả các phần tử <li> trong một danh sách
        //không sử dụng document.getElementsByTagName()
        var listItems = document.querySelectorAll("ul li");
        for (var j = 0; j < listItems.length; j++) {
            listItems[j].style.fontWeight = "bold";
        }
    </script>
</body>
```



Thay đổi nội dung và thuộc tính

- element.innerHTML
- element.innerText
- element.textContent
- element.style.css_property = value
- element.getAttribute(attribute_name)

Thay đổi nội dung và thuộc tính

element.innerHTML

- Đại diện cho nội dung bên trong một phần tử HTML.
- Cho phép truy cập và thay đổi nội dung HTML của một phần tử.

```
<body>
    <p id="myDiv">Công nghệ Thông tin</p>

    <script>
        var myDiv = document.getElementById("myDiv");

        myDiv.innerHTML = "CÔNG NGHỆ THÔNG TIN";
    </script>
</body>
```

Thay đổi nội dung và thuộc tính

element.innerText

- Đại diện cho văn bản (text) bên trong một phần tử HTML, bỏ qua bất kỳ thẻ HTML nào trong nội dung của phần tử.
- Trả về hoặc gán mới text không HTML của phần tử.

```
<body>
    <div id="myDiv"><strong>Toàn diện - Sáng tạo - Phụng sự</strong></div>

    <script>
        var myDiv = document.getElementById("myDiv");
        var textContent = myDiv.innerText;

        console.log(textContent);
    </script>
</body>
```

Thay đổi nội dung và thuộc tính

element.textContent

- Truy cập và thay đổi nội dung văn bản (text) bên trong phần tử.
- Trả về hoặc cập nhật nội dung văn bản (không HTML).

```
<body>
    <p id="pa">Lập trình Web</p>

    <script>
        var pa = document.getElementById("pa");

        var textContent = pa.textContent;
        console.log(textContent);

        // Gán lại nội dung văn bản mới cho phần tử
        pa.textContent = "Internet và Công nghệ Web";
    </script>
</body>
```

Thay đổi nội dung và thuộc tính

element.style.css_property = value

- Thay đổi kiểu dáng của một phần tử HTML bằng cách thay đổi giá trị của một thuộc tính CSS.
- element là tham chiếu đến phần tử HTML.
- css_property tên của thuộc tính CSS muốn thay đổi. Ví dụ: "color", "font-size", "background-color", và nhiều thuộc tính khác.
- value là giá trị muốn thiết lập cho thuộc tính CSS.

Thay đổi nội dung và thuộc tính

element.style.css_property = value

- Ví dụ

```
<body>
    <p id="myElement">Toàn diện - Sáng tạo - Phụng sự</p>

    <script>
        var myElement = document.getElementById("myElement");

        myElement.style.color = "red";
        myElement.style.fontSize = "20px";
        myElement.style.backgroundColor = "blue";
    </script>
</body>
```

Thay đổi nội dung và thuộc tính

element.getAttribute(attribute_name)

- Lấy giá trị của một thuộc tính cụ thể của một phần tử HTML
- attribute_name tên của thuộc tính muốn lấy giá trị. Ví dụ: "src", "href", "alt", "class", "id", và nhiều thuộc tính khác.

```
<body>
    <a id="myLink" href="http://uit.edu.vn">Xem trang</a>
    <script>
        var myLink = document.getElementById("myLink");
        var hrefValue = myLink.getAttribute("href");

        console.log(hrefValue);
    </script>
</body>
```

Thay đổi nội dung và thuộc tính

element.setAttribute(attribute_name, value)

- Thay đổi giá trị của một thuộc tính
- attribute_name tên thuộc tính muốn thiết lập hoặc thay đổi. Ví dụ: "src", "href", "alt", "class", "id", và nhiều thuộc tính khác.
- value giá trị mới muốn cập nhật cho thuộc tính.

```
<body>
    <img id="myImage" src="path.jpg" alt="Hình không tồn tại">

    <script>
        var myImage = document.getElementById("myImage");

        myImage.setAttribute("src", "new_path.jpg");
        myImage.setAttribute("alt", "Hình mới");
    </script>
</body>
```

Thay đổi nội dung và thuộc tính

element.removeAttribute(attribute_name)

- Xóa một thuộc tính cụ thể khỏi một phần tử HTML.
- Ví dụ

```
<body>
    <a id="myLink" href="http://uit.edu.vn">Xem trang</a>

    <script>
        var myLink = document.getElementById("myLink");
        myLink.removeAttribute("href");
    </script>
</body>
```

Sự kiện và xử lý sự kiện

- Có thể tạo sự kiện (event) cho một phần tử HTML và chỉ định một hàm xử lý sự kiện để thực thi khi sự kiện đó xảy ra.
- Tạo sự tương tác lên trang web, các phần tử tương tác với người dùng.
- Cú pháp

```
element.addEventListener(event, function, useCapture);
```

- event: Tên sự kiện, ví dụ: "click", "mouseover", "keydown"…
- function: Hàm xử lý sự kiện (callback function) thực thi khi sự kiện xảy ra.
- useCapture (tùy chọn): Giá trị boolean (true hoặc false: mặc định) xác định xem sự kiện dùng capture hay không.

Bắt sự kiện cách 1

```
<body>
    <button onclick="myFunction()">Click</button>

    <script>
        function myFunction() {
            alert("Nút đã được nhấn!");
        }
    </script>
</body>
```

Bắt sự kiện cách 2

```
<body>
    <button id="myButton">Click</button>

    <script>
        var myButton = document.getElementById("myButton");

        myButton.addEventListener("click", function () {
            alert("Nút đã được nhấn!");
        });
    </script>
</body>
```

Cách 1. Lấy giá trị từ các thẻ:

- <input> (text, radio button)
- <select>
- <textarea>
- <checkbox>

```
<input type="text" id="textInput" />

<select id="selectInput">
    <option value="option1">Lựa chọn 1</option>
    <option value="option2">Lựa chọn 2</option>
    <option value="option3">Lựa chọn 3</option>
</select>

<textarea id="textareaInput"></textarea>

<input type="checkbox" id="checkboxInput" />

<input type="radio" id="radioInput1" name="radioGroup" value="radio1" />
<input type="radio" id="radioInput2" name="radioGroup" value="radio2" />

<button id="btn">Lấy giá trị</button>

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

```
let btn = document.getElementById("btn");
btn.addEventListener("click", function(){
    var textInput = document.getElementById("textInput").value;
    var selectInput = document.getElementById("selectInput").value;
    var textareaInput = document.getElementById("textareaInput").value;
    var checkboxInput = document.getElementById("checkboxInput").checked;

    var radioInput1 = document.getElementById("radioInput1").checked;
    var radioInput2 = document.getElementById("radioInput2").checked;

    var result = "Giá trị input: " + textInput + "<br>";
    result += "Lựa chọn từ select: " + selectInput + "<br>";
    result += "Nội dung textarea: " + textareaInput + "<br>";
    result += "Trạng thái checkbox: " + checkboxInput + "<br>";

    result += "Trạng thái radio 1: " + radioInput1 + "<br>";
    result += "Trạng thái radio 2: " + radioInput2 + "<br>";

    document.getElementById("result").innerHTML = result;
});
```

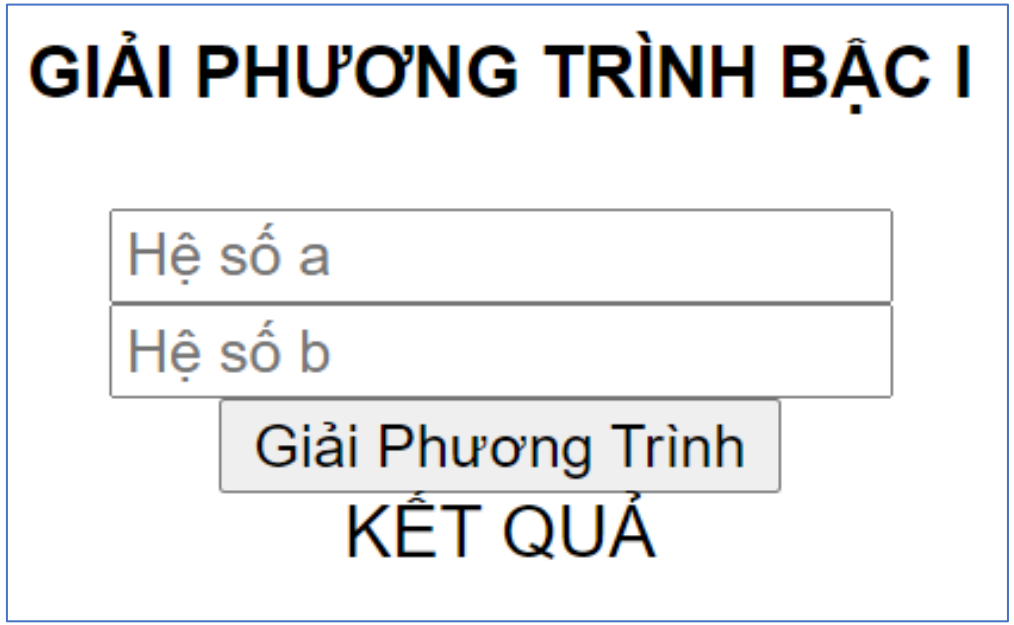
Cách 2. Lấy giá trị từ các thẻ:

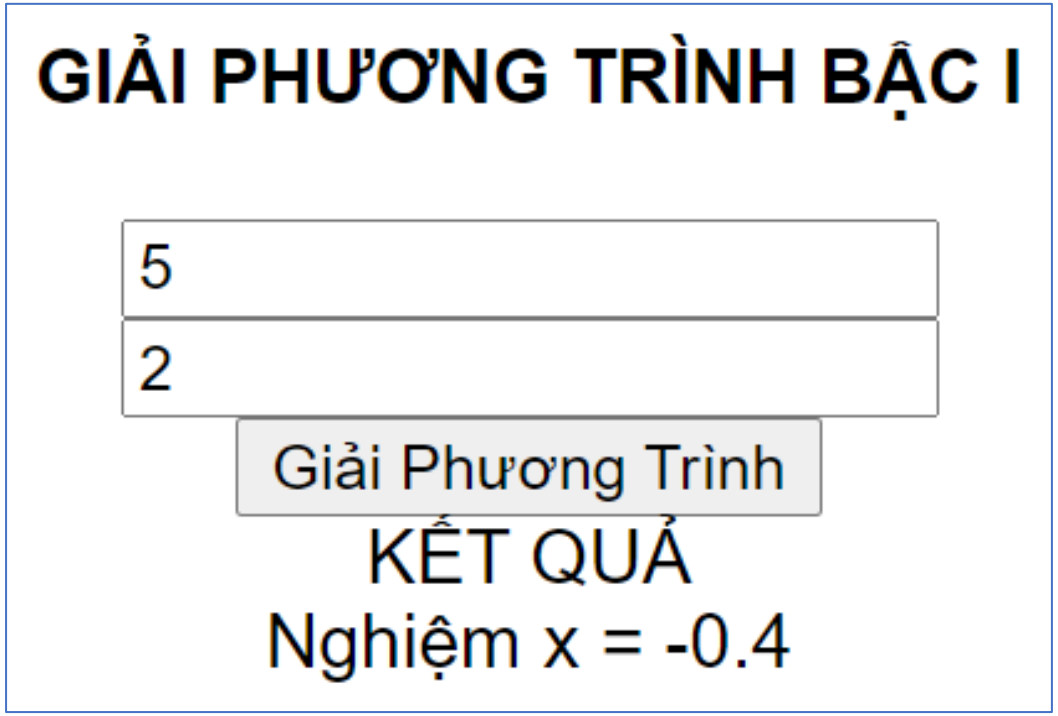
- <input> (text, radio button)
- <select>
- <textarea>
- <checkbox>

```
function getValues() {
    var textInput = document.getElementById("textInput").value;
    var selectInput = document.getElementById("selectInput").value;
    var textareaInput = document.getElementById("textareaInput").value;
    var checkboxInput = document.getElementById("checkboxInput").checked;

    var radioInput1 = document.getElementById("radioInput1").checked;
    var radioInput2 = document.getElementById("radioInput2").checked;

    var result = "Giá trị input: " + textInput + "<br>";
    result += "Lựa chọn từ select: " + selectInput + "<br>";
    result += "Nội dung textarea: " + textareaInput + "<br>";
    result += "Trạng thái checkbox: " + checkboxInput + "<br>";

    result += "Trạng thái radio 1: " + radioInput1 + "<br>";
    result += "Trạng thái radio 2: " + radioInput2 + "<br>";

    document.getElementById("result").innerHTML = result;
}
```

```
<input type="text" id="textInput" />

<select id="selectInput">
    <option value="option1">Lựa chọn 1</option>
    <option value="option2">Lựa chọn 2</option>
    <option value="option3">Lựa chọn 3</option>
</select>

<textarea id="textareaInput"></textarea>

<input type="checkbox" id="checkboxInput" />

<input type="radio" id="radioInput1" name="radioGroup" value="radio1" />
<input type="radio" id="radioInput2" name="radioGroup" value="radio2" />

<button onclick="getValues()">Lấy giá trị</button>
```

Bài tập


GIẢI PHƯƠNG TRÌNH BẬC I
Hệ số a
Hệ số b
Giải Phương Trình
KẾT QUẢ


GIẢI PHƯƠNG TRÌNH BẬC I
5
2
Giải Phương Trình
KẾT QUẢ
Nghiệm x = -0.4

Bài tập

Phiếu đăng ký tham gia chương trình khuyến mãi

Họ tên	
Địa chỉ	
Điện thoại	
Giới tính	Nam Nu
Nghề nghiệp	Giáo viên
Chọn sản phẩm tham gia	Kem đánh răng Bột giặt Dầu gội đầu Sữa tắm
Số người dự đoán tham gia	300000
Đồng ý	
Bạn đã đăng ký thành công!!!	

- Sử dụng kiểu object
- Thay đổi className


Thank you

Thảo luận