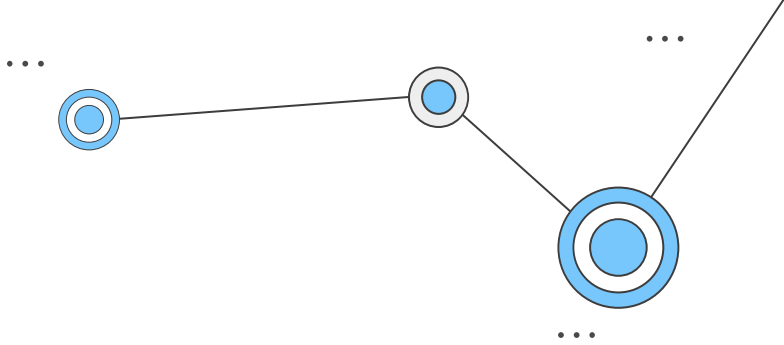
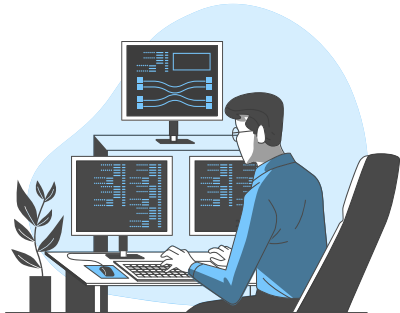
JS

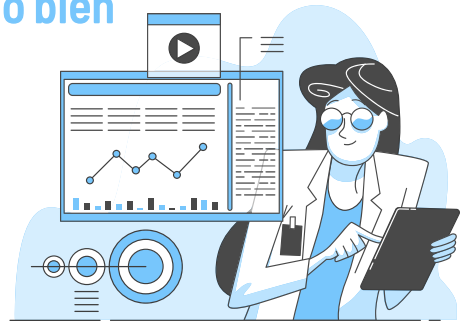


Khóa học Backend

Bài 06: Javascript cơ bản (Tiết 2)



Nội dung



01

Câu lệnh rẽ nhánh
If Else

1.1. Câu lệnh if

- Câu lệnh if(...) sẽ **kiểm tra điều kiện** biểu thức bên **trong cặp dấu ngoặc đơn ()**.
- Nếu kết quả là **true** thì một khối code sẽ **được thực thi**.
- Cú pháp:

```
if (condition) {
  //nếu điều kiện đúng thì thực hiện
}
```

- Trong đó: **condition** (điều kiện) là mệnh đề điều kiện và luôn luôn phải có một trong hai giá trị là true/false, hoặc giá trị tương đương như:
  - NULL => false
  - Rỗng "" => false
  - Số khác 0 => true
  - Số 0 => false

1.2. Lệnh if else

- Lệnh else sẽ được thực thi nếu lệnh if không được thực hiện, tức là điều kiện ở condition sẽ có giá trị là false.
- Lưu ý: khi dùng lệnh else thì bắt buộc phải có một lệnh if đứng trước nó.
- Cú pháp:

```
if (condition) {
  // Code cho lệnh if
} else {
  // Code cho lệnh else
}
```

1.3. Kết hợp nhiều lệnh if else

- Ta có thể kết hợp nhiều câu lệnh if để xử lý bài toán, bằng cách thêm nhiều lệnh else if phía sau.
- Cú pháp:

```
if (condition1) {
  // Code cho lệnh if 1
} else if (condition2) {
  // Code cho lệnh if 2
} else {
  // Code cho lệnh else
}
```

1.4. Toán tử 3 ngôi

- **Rút gọn** câu lệnh **if else** bằng cách dùng toán tử rẽ nhánh **?**.
- Cú pháp: var result = **condition ? value1 : value2**;
- Đây là toán tử **ba ngôi** với ba thành phần:
  - **condition**: điều kiện cần kiểm tra
  - **value1**: trả về giá trị value1 nếu điều kiện **condition** là **true**.
  - **value2**: trả về giá trị value2 nếu điều kiện **condition** là **false**.

02

Switch Case

02. Switch Case

- **Dùng để rẽ nhánh** chương trình.
- **Mỗi nhánh** sẽ **có một điều kiện** cụ thể và **điều kiện** đó phải sử dụng toán tử **so sánh bằng**.
- Cú pháp:

```
switch (variable) {
  case value_1:
    // do some thing
    break;
  case value_2:
    // do some thing
    break;
  default:
    // do something
    break;
}
```

02. Switch Case

- Trường hợp **bỏ break** đi thì câu lệnh sẽ **chạy tiếp xuống case tiếp theo**, đến case nào **có break thì dừng lại**.
- Trường hợp **không có default** thì nếu **rơi vào trường hợp này sẽ không in ra gì**.
- Ta cũng có thể **gom nhóm case** lại.

03

Vòng lặp For

03. Vòng lặp For

- Vòng lặp **for** dùng để **lặp lại** việc thực thi **một đoạn mã** nào đó **một số lần**.
- Cú pháp:

```
for (bienkhoitao; dieukienthucthi; buocnhay) {
  // code...
}
```

- Trong đó:
  - **bienkhoitao**: là **giá trị khởi tạo** ban đầu của vòng lặp.
  - **dieukienthucthi**: là **điều kiện** mà vòng lặp được phép chạy (chú ý: Nếu bạn bỏ trống sẽ bị lặp vô tận).
  - **buocnhay**: là **khoảng đệm nhảy** của mỗi vòng lặp.

04

Vòng lặp For In

04. Vòng lặp For In

- Vòng lặp **for in** dùng để **lấy ra key** của một **object** (cũng có thể áp dụng cho **array, string**).
- Cú pháp:

```
for (key in object) {
  // Viết code ở đây
}
```

- Trong đó:
  - **key**: là **tên biến** (đặt tên là gì cũng được).
  - **object**: là một object, **mỗi key** sẽ tương ứng với **key ở trong object**.

05

Vòng lặp For Of

05. Vòng lặp For Of

- Vòng lặp **for of** dùng để lấy ra phần tử của một **array, string**.
- Không áp dụng được với object.
- Cú pháp:

```
for (variable of iterable) {
  // Viết code ở đây
}
```

- Trong đó:
  - **variable**: là **tên biến**, lấy ra các giá trị trong mảng.
  - **iterable**: là một **mảng** hoặc **string**, mỗi variable sẽ tương ứng với từng giá trị ở trong mảng hoặc string.

06

Vòng lặp While

6.1. Vòng lặp while

- Vòng lặp **while luôn luôn kiểm tra điều kiện trước**, nếu điều kiện **thỏa mãn** thì **mới chạy** vào trong vòng lặp.
- Cú pháp:

```
while (condition) {
  // code
}
```

- Trong đó:
  - **condition**: là **điều kiện** thực hiện vòng lặp.
  - Khi **condition** có giá trị **true** thì **code được thực thi**.
  - Ngược lại, khi **condition** là **false** thì **vòng lặp kết thúc**.

6.2. Vòng lặp do...while

- Vòng lặp **do...while luôn thực hiện ít nhất một lượt lặp**.
- Sau đó mới kiểm tra điều kiện lặp.
- Cú pháp:

```
do {
  // code
} while (condition);
```

- Trong đó:
  - **condition**: là **điều kiện** thực hiện vòng lặp.
  - Khi **condition** có giá trị **true** thì **code được thực thi**.
  - Ngược lại, khi **condition** là **false** thì **vòng lặp kết thúc**.

07

Break

07. Break

- Lệnh **break** có tác dụng **dừng vòng lặp** cho dù điều kiện của vòng lặp vẫn đang đúng.
- Cú pháp:

```
break;
```

08

Continue

08. Continue

- Lệnh **continue** có tác dụng **bỏ qua một bước lặp** nào đó.
- Nghĩa là lúc **gặp lệnh continue** thì tất cả những đoạn code nằm bên dưới sẽ không được thực hiện mà nó sẽ **nhảy sang vòng lặp** mới luôn.
- Cú pháp:

```
continue;
```

09

Các cách khai báo biến

09. Các cách khai báo biến

- Từ khóa **var**
  - Phạm vị truy cập toàn cục.
  - Cú pháp: **var** tenBien = giaTri;

  - Trong đó:
    - **tenBien**: Là tên của biến các bạn muốn đặt.
    - **giaTri**: Là giá trị của biến, có thể là số, chuỗi, mảng, object.

09. Các cách khai báo biến

- Từ khóa **let**
  - Chỉ có phạm vi trong khối khai báo.
  - Cú pháp: **let** tenBien = giaTri;

09. Các cách khai báo biến

- Từ khóa **const**
  - Chỉ có phạm vi trong khối khai báo.
  - Là **hằng số** nên **không thể gán lại giá trị**.
  - Cú pháp: **const** tenBien = giaTri;

09. Các cách khai báo biến

- Khai báo biến **không cần dùng từ khóa**
  - Chỉ nên sử dụng cách này khi muốn **gán lại giá trị của một biến**.
  - Cú pháp: tenBien = giaTri;

Bài tập

- **Link bài tập**: https://dacavn.notion.site/B-i-t-p-b-i-06-Javascript-c-b-n-Ti-t-2-4dae639abf4e4747b06c1f8a3e99c172?pvs=4