Content

[I.HTML+CSS 2](#_Toc110064538)

[1.CSS selectors 2](#_Toc110064539)

[2.Priority—ưu tiên áp dụng 3](#_Toc110064540)

[3.Units - đơn vị đo lường 3](#_Toc110064541)

[4.CSS-Variable—biến 4](#_Toc110064542)

[5.Box-sizing-giữ nguyên kích thước element 5](#_Toc110064543)

[6.Background-clip—đổ màu nền từ đâu vào 5](#_Toc110064544)

[7. Background-image—đặt ảnh nền 5](#_Toc110064545)

[8.Background-size – kích thứơc ảnh 6](#_Toc110064546)

[9.Background-origin – đổ màu ảnh nền 6](#_Toc110064547)

[10. Background-position – tùy chỉnh vị trí ảnh nền 6](#_Toc110064548)

[11.Background shorthand 6](#_Toc110064549)

[12.Function - hàm 7](#_Toc110064550)

[attr() – attribute-thuộc tính 7](#_Toc110064551)

[13.Pseudo classes – lớp giả 7](#_Toc110064552)

[14.Bold – in đậm 7](#_Toc110064553)

[15.Flexbox 8](#_Toc110064554)

[II.Javascript 9](#_Toc110064555)

# I.HTML+CSS

## 1.CSS selectors

Trong bài này mình chỉ đưa ra những CSS selectors cơ bản và thường xuyên được sử dụng nhất.

|  |  |  |
| --- | --- | --- |
| **Selector** | **Ví dụ** | **Mô tả** |
| .class | .intro | Chọn tất cả các thẻ có class=“intro” |
| .class1.class2 | .name1.name2 | Chọn tất cả các thẻ có cả name1 và name2 được đặt trong thuộc tính class của nó |
| .class1 .class2 | .name1 .name2 | Chọn tất cả các thẻ có class=“name2” là con của một phần tử có class=“name1” |
| \* | \* | Chọn tất cả các thẻ |
| element | h2 | Chọn tất cả các thẻ h2 |
| element.class | div.box | Chọn tất cả thẻ div có class=“box” |
| element, element | div, h2 | Chọn tất cả thẻ div và h2 |
| element element | div p | Chọn tất cả thẻ p trong thẻ div |
| element > element | div > p | Chọn tất cả thẻ p là con trực tiếp của thẻ div |
| element + element | div + p | Chọn thẻ p đứng liền kề sau thẻ div |
| element ~ element | div ~ p | Chọn tất cả thẻ p đứng sau thẻ div |

## 2.Priority—ưu tiên áp dụng

2.1. Internal, External

- Cái nào gọi sau thì đựơc áp dụng

2.2. inline - 1000

2.3. #id - 100

2.4.class - 10

2.5. tag - 1

- tag - gọi thẳng tên của thẻ

h1 {

}

2.6. equal specificity?

- Các thẻ giống nhau được gọi lại nhiều lần

2.7. universal selector and inherited? - 0

## 3.Units - đơn vị đo lường

(Để tùy chỉnh kích thước hiển thị)

3.1.Absolute units\_tuyệt đối --- không đổi

\*px – điểm ảnh, độ phân giải

\*pt

\*cm

\*mm

\*inch

\*pc

3.2.Relative units\_tương đối----vs cái gì

Bị thay đổi

\*%

-Phụ thuộc thẻ chứa nó

- Thường sử dụng khi chia layout

\*rem

-Phụ thuộc thẻ html{ }

1rem = …px trong định dạng css cho thẻ html

🡪dễ kiểm soát vì chỉ cần quan tâm tới thẻ html

Nên đặt font-size thẻ html = 100% tương đương 16px

Ưu điểm khi sử dụng rem cho font-size

\*em

-Phụ thuộc thẻ gần nhất chứa nó, mà chứa thuộc tính có đơn vị là em

Vd thuộc tính font-size: 2em;

🡪tìm thẻ cha có thuộc tính font-size gần nó nhất

🡪khó kiểm soát

\*vw – viewport width chiều ngang của trình duyệt

Viewport – khung nhìn, độ rộng trình duyệt

50vw 50% chiều ngang của trình duyệt

50vw != 50%

\*vh - viewport height chiều dài của trình duyệt

🡪khi nào muốn 1 đối tượng có chiều rộng và chiều cao luôn bằng vs màn hình🡪sử dụng thuộc tính vw và vh

\*vmin

\*vmax

\*ex

\*ch

## 4.CSS-Variable—biến

-Sử dụng lớp giả🡪tạo biến toàn cục🡪sử dụng mọi nơi

Vd

/\*Khai báo biến\*/

: root {

--text-color: red;

}

h1 {

color: var(--text-color);

}

var(…) sử dụng hàm

Tạo biến local:

h1 {

--my-color: green;

color:var(--my-color);

}

## 5.Box-sizing-giữ nguyên kích thước element

ứng dụng: giữ nguyên kích thước đối tượng khi padding và border biến động

box-sizing: border-box;

## 6.Background-clip—đổ màu nền từ đâu vào

- Áp dụng cho background-color

## 7. Background-image—đặt ảnh nền

Hình ảnh nào viết trước sẽ ở trên:

background-image:  url(…), url(…);

kích thước ảnh nền:

+ chiều rộng 100%

+ chiều cao auto

background-size: 100% auto;

lặp lại hình ảnh:

background-repeat: no-repeat;

repeat-y

repeat-x

Tạo ra dải màu chuyển đều đặn

linear-gradient(0deg, #333, #ccc)

hướng xoay : 0deg

màu bắt đầu chuyển: #333\_màu đặc

màu kết thúc chuyển: #ccc

Tạo màu trong suốt: rgba(255, 255, 255, 0.2)

a-độ trong suốt giảm dần 0-1

## 8.Background-size – Kích thước ảnh

Cố gắng lấy chiều dài hơn (của trang web)

+Điều kiện: ko cắt xén, che khuất bức ảnh

+Có khoảng trắng

Background-size: contain;

Cố gắng lấy chiều dài hơn (của trang web)

+Chấp nhận che khuất bức ảnh

+ko có khoảng trắng

Background-size: cover;

## 9.Background-origin – đổ màu ảnh nền

- Tương tự như background-clip—nhưng áp dụng cho background-image

+ Mặc định là ảnh đổ từ lớp padding-box

Background-origin: content-box;

## 10. Background-position – tùy chỉnh vị trí ảnh nền

background-position: top right;

Viết 1 giá trị, giá trị thứ 2 tự động là center

background-position: 50px 30px;

chiều rộng: cách lề trái 50px

chiều cao: cách lề trên 30px

## 11.Background shorthand

Background image, repeat, position, size.

Background: #333 url(…) no-repeat center / contain

Màu, link ảnh,lặp lại ko, căn chỉnh vị trí, kích thức

Cần thêm 1 dấu gạch chéo trc background-size

No-repeat đưa vào ngay sao link hình ảnh

## 12.Function - hàm

Hàm rgb() - tạo ra màu đặc — ko thể nhìn xuyên qua

calc() – tính toán

Ứng dụng trong cộng cả hai giá trị tương đối và tuyệt đối cùng 1 lúc.

### attr() – attribute-thuộc tính

thường được dùng với lớp giả

 <a href="https://fullstack.edu.com.vn/">Học lập trình f8 </a>

/\* lớp giả \*/

a::after {

    content: "(" attr(href)  ")";

}

## 13.Pseudo classes – su đồ - lớp giả

Example:

h1:hover {

}

- :hover - di mouse into will have change (or active)

- :active - press and keep mouse 🡪 will have change

- :first-child / :last-child - apply for tag <ul> and <li>

## 14.CSS - Pseudo element-phần tử giả

Tác dụng: tạo ra các phần tử hiển thị trên web site mà không dung html code, chỉ cần css code thôi

Chú ý: mỗi 1 phần tử giả chỉ tồn tại 1 lần; css bình thường

::before

Vd: vẽ 1 hình vuông trong 1 thẻ div

.box::before {

Content: “”; 🡪 giúp thẻ tồn tại, không có không được, cho dù là chuỗi rỗng

Display: block; 🡪 tạo khối

Width: 50px;

Height: 50px;

Background-color: while;

}

::after tương tự ::before

::first-letter

Làm to chữ cái đầu tiên (các bài báo ngày xưa)

h1 :: first-letter{

font-size: 50px;

}

::first-line

::selection

-khi bôi đen sẽ có sự thay đổi trên web site

15.Relative – giao diện đè lên nhau - ứng dụng cho thông báo, tiến độ khoá học…

Thiết lập vị trí hiện thị của phần tử trên web site

Position:

+relative – tương đối

Lấy chính mình làm gốc toạ độ, không bị phụ thuộc vào bất kỳ đối tượng nào cả.

h1{

Position: relative; (không có Position; top, right không hoạt động)

Top: 100px;

Right: 100px;

}

+absolte-vị trí tuyệt đối

Phụ thộc thẻ cha gần nhất có thuộc tính position. Lấy thẻ cha là gốc toạ độ

Vd:

<div class="box">

<div class="box-child"></div>

</div>

.box{

width: 100%;

height: 120px;

background-color: #333;

position: relative; 🡪 1 trong 4 thuộc tính(relative, absolute, fixed, sticked) không quan trọng. 🡪 quan trọng là keyword “position”

}

.box-child{

width: 50px;

height: 50px;

background-color: #fff;

position: absolute;

right: 0;

bottom:0;

//chỉ sử dụng được 2 / 4 🡪 ưu tiên left, top khi viết cả 4

//bottom: 0;

//left: 0;

}

🡪 các thẻ div lồng nhau 🡪 tạo hiệu ứng khối chứa khối (không giới hạn số lượng khối child1, child2, child3, child4….)

+fixed – phụ thuộc khung trình duyệt

Vd: phần header giữ nguyên khi kéo trình duyệt

.header {

Position: fixed;

Top: 0; //header trên

// or bottom: 0; //header dưới

Width: 100%;

Hight: 34px;

}

+sticky – bám dính- phụ thuộc khung trình duyệt (khuyến cáo không sử dụng)🡪 thay thế bằng fixed+javascript

ứng dụng trong Menu, header, navbar….

Kiểm tra tag nào được trình duyệt nào hỗ trợ:

Google search keyword “caniuse.com”

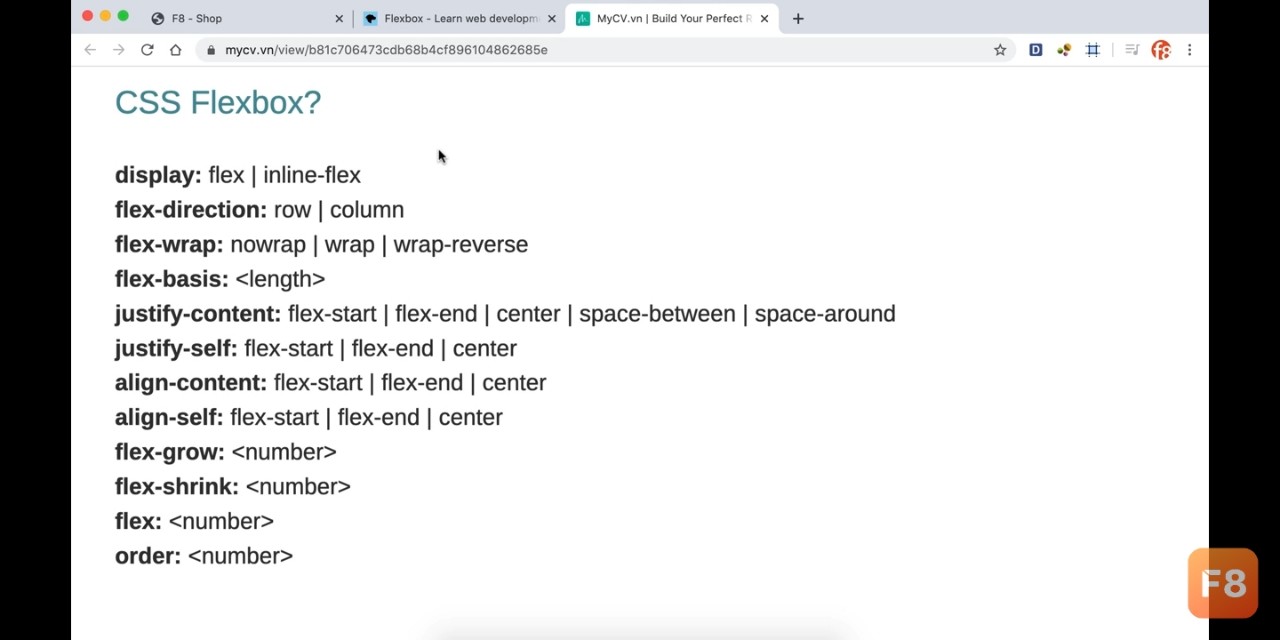
## 15.Bold – in đậm

Tag <b> in đậm văn bản

Tag <i> in nghiêng

Tag <u> gạch chân

## 15.Flexbox



Display: flex; 🡪 flex-direction mặc định là : row;

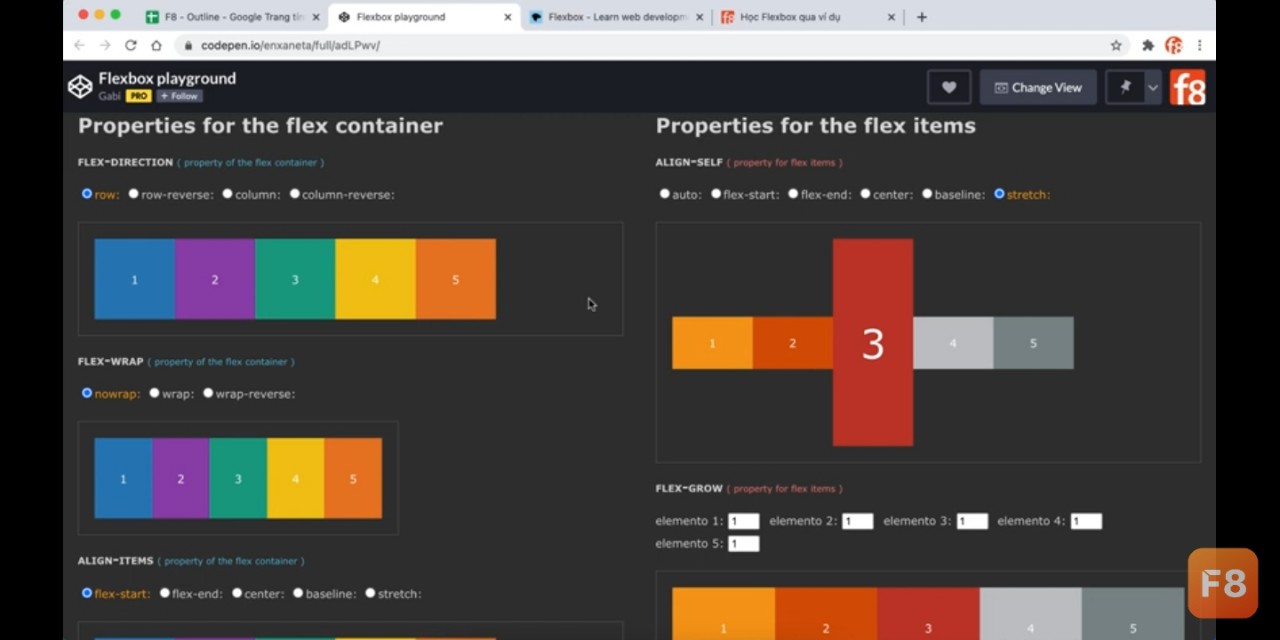
-justify-content:

+ space-around: ngược lại với space-between 🡪 có khoảng trống ở 2 bên

+ space-between: không có khoảng trống ở 2 bên (item ngoài cùng bên trái và bên phải

+ space-evenly: tất cả các khoảng cách đều bằng nhau.

Luyện tập flexbox



## 16. Navigation Bar – thanh điều hướng

## 17.CSS BEM

# BEM

- Là tiêu chuẩn đặt tên class khi viết CSS

## Ý nghĩa

- Viết tắt của: block element modifier

- Block: khối

- Element: thành phần trong khối

- Modifier: bổ xung ý nghĩa cho ‘Block’ hoặc ‘Element’

## Tại sao phải dung BEM

- Mỗi người đặt một kiểu

- Members đặt class trùng nhau, CSS đè lên nhau

## Cú pháp

- .block

- .block\_\_element

- .block--modifier

- . block\_\_element--modifier

## Tính ứng dụng

- Xây dựng layout website

- Xây dựng thành phần trên website

## Ưu điểm

- Tính rõ rang

- Tái sử dụng dễ dàng

- Giúp cả team làm việc với nhau dễ dàng

- Tính module, không lo CSS của class này ảnh hưởng lên CSS của class khác

## Nhược điểm

- Tên class dài

- Một số người cho là xấu

## Khi nào dung BEM là phù hợp?

- Dự án nhiều members

- Dự án lớn, số lượng pages nhiều hoặc số lượng các thành phần trên giao diện nhiều

## Thực hành

- Làm button

- Làm message

- Làm 1 thành phần trên website

# II.Javascript

Kiểu Boolean

Phía trc biến có is

Vd: isSuccess

Kiểm tra kiểu dữ liệu biến:

Console.log(typeof values);

II.10.HTML DOM