Khóa học HTML và CSS cơ bản: <https://fullstack.edu.vn/courses/html-css>

|  |  |  |
| --- | --- | --- |
| .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 |

Cách đặt biến và sử dụng:

:root{

--text-color:red;

}

Ví dụ cho thể p có màu

p{

color:var(--text-color);

}

Các đơn vị trong css thường dùng:

* px
* %
* rem
* vw
* vh

các hàm trong css thường dùng:

**var()**

**linear-gradient()**

Tạo màu nền dải màu tuyến tính (gradient).

Ví dụ: background: linear-gradient(hướng, màu, màu);

to bottom: Tạo gradient từ trên xuống dưới.

background: linear-gradient(to bottom, red, blue);

to top: Tạo gradient từ dưới lên trên.

background: linear-gradient(to top, red, blue);

to left: Tạo gradient từ phải sang trái.

background: linear-gradient(to left, red, blue);

to bottom right: Tạo gradient từ góc trên trái xuống góc dưới phải.

background: linear-gradient(to bottom right, red, blue);

to top left: Tạo gradient từ góc dưới phải lên góc trên trái.

background: linear-gradient(to top left, red, blue);

**rgba()**

rgba(255, 0, 0, 0.5)

R (Red): Đỏ. Giá trị 255 cho biết màu đỏ là tối đa.

G (Green): Xanh lá cây. Giá trị 0 cho biết không có màu xanh lá.

B (Blue): Xanh dương. Giá trị 0 cho biết không có màu xanh dương.

A (Alpha): Độ trong suốt. Giá trị 0.5 có nghĩa là màu này trong suốt 50%.

**rgb()**

**Tương tự như rgba() nhưng không có giá trị alpha.**

**Ví dụ: color: rgb(255, 0, 0); (màu đỏ không trong suốt)**

**calc()**

là một hàm tính toán cho phép bạn thực hiện các phép toán như cộng, trừ, nhân, và chia với các đơn vị khác nhau trong CSS.

width: calc(100% - 50px);

Trong ví dụ này:

* 100%: Chiều rộng của phần tử cha.
* - 50px: Trừ đi 50 pixel.

Kết quả là phần tử sẽ có chiều rộng bằng 100% chiều rộng của phần tử cha trừ đi 50 pixel.

**attr()**

là một hàm trong CSS dùng để truy xuất giá trị của một thuộc tính trên phần tử HTML.

attr(name)

Lấy giá trị của thuộc tính name của phần tử

Css:

p::after { content: attr(data-content)};

* p::after: Chỉ định phần tử p và sử dụng pseudo-element ::after.
* content: attr(data-content);: Sẽ chèn nội dung của thuộc tính data-content của phần tử p vào vị trí của pseudo-element ::after.

Html

<p data-content="Hello, World!"></p>

Các lớp giả trong css (**Pseudo classes trong CSS**)

:root

:hover Áp dụng khi người dùng đưa con trỏ chuột lên phần tử

:active Áp dụng khi phần tử đang được người dùng nhấn xuống

:first-child Chọn phần tử con đầu tiên của phần tử cha

:last-child Chọn phần tử con cuối cùng của phần tử cha

:nth-child() Chọn một phần tử dựa trên vị trí của nó trong danh sách các phần tử con của phần tử cha.

ví dụ cho :first-child và :last-child

<div>

<p>Đoạn văn đầu tiên</p>

<p>Đoạn văn giữa 2</p>

<p>Đoạn văn giữa 3</p>

<p>Đoạn văn giữa 4</p>

<p>Đoạn văn cuối cùng</p>

</div>

div p:first-child {

color: blue;

}

div p:last-child {

color: red;

}

Chọn phần tử con thứ 2

div p:nth-child(2) {

color: red;

}

:nth-child(old) chọn các phần tử lẻ 1,3,5,7,...

:nth-child(even) chọn các phần tử chẵn 2,4,6,8,...

**Pseudo elements trong CSS**

**::before** Thêm nội dung trước nội dung của phần tử. Thường dùng để chèn biểu tượng hoặc văn bản.

**::after** Thêm nội dung sau nội dung của phần tử. Thường dùng để thêm các yếu tố trang trí hoặc ký hiệu.

**::selection** Áp dụng kiểu cho văn bản được người dùng chọn.

<p>Hello, world!</p>

p::before {

content: "🔥 ";

}

p::after {

content: " 🌍";

}

p::selection {

background: yellow;

color: black;

}

::before thêm biểu tượng lửa trước văn bản "Hello, world!".

::after thêm biểu tượng trái đất sau văn bản.

::selection thay đổi nền và màu chữ khi văn bản được chọn.

Văn bản được chọn là phần văn bản mà người dùng đánh dấu để chọn, thường bằng cách kéo chuột hoặc dùng bàn phím. Ví dụ cụ thể là click chuột và kéo trong lúc đó không buôn click ra

**Background-image**

là một thuộc tính cho phép bạn đặt hình ảnh làm nền cho một phần tử.

background-repeat: Quyết định xem hình nền có lặp lại hay không. Giá trị có thể là repeat, no-repeat, repeat-x, repeat-y.

background-size: Quy định kích thước của hình nền. Giá trị có thể là cover, contain, hoặc kích thước cụ thể như 100px 200px.

background-position: Quy định vị trí của hình nền. Giá trị có thể là left, right, center, top, bottom, hoặc các giá trị cụ thể như 10px 20px.

background-attachment: Quyết định xem hình nền có cuộn theo phần tử hay không. Giá trị có thể là scroll, fixed, hoặc local.

Cú pháp:

background-image: url('background.jpg');

background-image: url('background.jpg') , url('background.jpg');

Cú pháp rút gọn (shorthand) cho thuộc tính background

selector {

background: [background-color] [background-image] [background-repeat] [background-attachment] [background-position] / [background-size];

}

div {

background: #ff0000 url('image.jpg') no-repeat fixed center / cover;

}

**Position**

Position:relative

Là một thuộc tính cho phép bạn định vị một phần tử tương đối so với vị trí ban đầu của nó. Điều này có nghĩa là phần tử vẫn chiếm không gian tại vị trí ban đầu trong tài liệu, nhưng bạn có thể di chuyển nó bằng cách sử dụng các thuộc tính top, right, bottom, và left.

.element {

position: relative;

top: 10px; /\* Di chuyển phần tử xuống 10px so với vị trí ban đầu \*/

left: 20px; /\* Di chuyển phần tử sang phải 20px so với vị trí ban đầu \*/

}

Position:absolute

là một thuộc tính cho phép bạn định vị một phần tử so với phần tử tổ tiên gần nhất có thuộc tính position

.container {

position: relative; /\* Phần tử tổ tiên chứa \*/

width: 300px;

height: 300px;

border: 1px solid black;

}

.element {

position: absolute;

top: 10px; /\* 10px từ cạnh trên của .container \*/

left: 20px; /\* 20px từ cạnh trái của .container \*/

}

Với position: absolute, phần tử sẽ không ảnh hưởng đến vị trí của các phần tử xung quanh nó và có thể di chuyển tự do trong khu vực của phần tử tổ tiên chứa.

Position:fixed

là một thuộc tính cho phép bạn định vị phần tử cố định so với cửa sổ trình duyệt. Điều này có nghĩa là phần tử sẽ giữ nguyên vị trí trên màn hình khi bạn cuộn trang, không phụ thuộc vào các phần tử khác trong tài liệu.

.fixed-element {

position: fixed;

top: 10px; /\* 10px từ cạnh trên của cửa sổ trình duyệt \*/

right: 20px; /\* 20px từ cạnh phải của cửa sổ trình duyệt \*/

width: 200px; /\* Chiều rộng của phần tử \*/

height: 100px; /\* Chiều cao của phần tử \*/

background-color: lightblue; /\* Màu nền của phần tử \*/

}

Phần tử với position: fixed sẽ luôn xuất hiện ở vị trí được chỉ định, ngay cả khi bạn cuộn trang lên hay xuống.

Ví dụ thực tế có thể thấy ở các thanh điều hướng cố định ở đầu hoặc cuối trang web, hay các nút "Back to Top" luôn hiển thị ở góc dưới cùng của trang.

Position:sticky

là một thuộc tính cho phép một phần tử "dính" (sticky) vào vị trí cụ thể khi bạn cuộn trang. Một phần tử với position: sticky sẽ hoạt động như relative khi nó nằm trong vùng hiển thị và chuyển sang fixed khi nó đạt đến vị trí xác định.

.sticky-element {

position: sticky;

top: 0; /\* Dính khi phần tử đạt đến 0px từ cạnh trên của phần tử chứa \*/

background-color: yellow;

padding: 10px;

border: 1px solid black;

}

Khi cuộn trang, phần tử sticky-element sẽ dính vào cạnh trên của phần tử chứa và giữ nguyên vị trí đó cho đến khi phần tử chứa kết thúc.