

## BÀI 4: BOX MODEL VÀ CSS NÂNG CAO

### 4.1 Pseudo-classes for Links

Một thành phần rất quan trọng trong mọi website chính là liên kết. Cũng như một đối tượng văn bản thông thường, hoàn toàn có thể áp dụng các thuộc tính định dạng đã học ở 2 bài trước như định font chữ, gạch chân, màu chữ,... cho một liên kết. Hơn nữa, CSS còn cung cấp một điều khiển đặc biệt được gọi là pseudo-classes. Pseudo-classes cho phép bạn xác định các hiệu ứng định dạng cho một đối tượng liên kết ở một trạng thái xác định như khi liên kết chưa được thăm (a:link), khi rê chuột lên liên kết (a:hover), khi liên kết được thăm (a:visited) hay khi liên kết đang được kích hoạt - đang giữ nhấn chuột (a:active). Với điều khiển pseudo-classes cùng với các thuộc tính CSS đã học chắc chắn sẽ mang lại rất nhiều ý tưởng về trang trí liên kết cho trang web. Sau đây, chúng ta sẽ tiến hành một số ví dụ để tìm hiểu thêm về các khả năng trang trí cho một liên kết dựa trên pseudo-classes.

*Ví dụ 1:* Ví dụ này sẽ áp dụng 4 màu sắc khác nhau cho từng trạng thái liên kết: các liên kết chưa thăm có màu xanh lá; các liên kết mouse over sẽ có màu đỏ tươi; các liên kết đã thăm sẽ có màu đỏ và các liên kết đang kích hoạt có màu tím.

```
a:link {color:#00FF00}
a:hover {color:#FF00FF}
a:visited {color:#FF0000}
a:active {color:# 662D91}
```

*Ví dụ 2:* Tạo các hiệu ứng tương ứng với liên kết: các liên kết chưa thăm có màu xanh lá, kích cỡ font 14px; liên kết mouse over có màu đỏ tươi, kích cỡ font 1.2em, hiệu ứng nhấp nháy; liên kết đã thăm sẽ có màu xanh da trời, không có đường gạch chân; các liên kết đang kích hoạt có màu tím và font dạng small-caps.

```
a:link {
    color:#00FF00;
    font-size:14px
}
a:hover {
    color:#FF00FF;
    font-size:1.2em;
    text-decoration:blink
}
a:visited {
    color:#FF0000;
    text-decoration:none
}
a:active {
```

```
color:# 662D91;
font-variant:small-caps
}
```

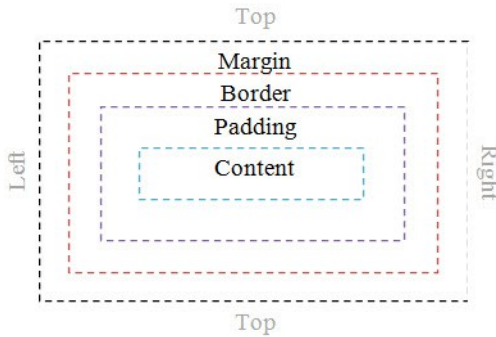
*Ví dụ 3:* Ví dụ này cũng tạo cho liên kết hiệu ứng màu sắc giống ví dụ 2 nhưng sẽ có thêm 1 số hiệu ứng: các liên kết sẽ có khung viền màu đen, kích cỡ font 14px; liên kết mouse over có nền light cyan; các liên kết đã thăm có nền light yellow.

```
a {
    border:1px solid #000;
    font-size:14px
}
a:link {
    color:#00FF00;
}
a:hover {
    background-color:#00BFF3;
    color:#FF00FF;
    font-size:1.2em;
    text-decoration:blink
}
a:visited {
    background-color:#FFF568;
    color:#FF0000;
    text-decoration:none
}
a:active { color:#662D91; font-variant:small-caps }
```

## 4.2 Box Model

Trong CSS, box model (mô hình hộp) mô tả cách mà CSS định dạng khối không gian bao quanh một thành phần. Nó bao gồm padding (vùng đệm), border (viền) và margin (canh lề) và các tùy chọn. Hình bên dưới mô tả cấu trúc minh họa mô hình hộp cho một thành phần web.

Mô hình hộp trên chỉ là một mô hình lý thuyết lý tưởng. Bên dưới đây, chúng ta sẽ xét mô hình hộp của một đối tượng web cụ thể:



Hình 16. Model Box

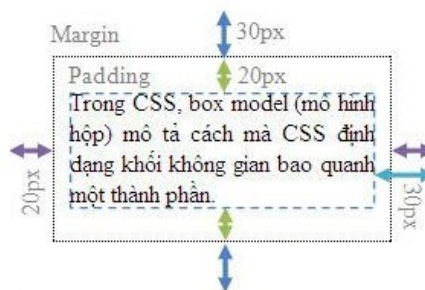
Chúng ta có một đoạn HTML sau:

<p>Trong CSS, box model (mô hình hộp) mô tả cách mà CSS định dạng khối không gian bao quanh một thành phần. </p>

Phần CSS cho đoạn HTML trên:

```
p {
    width:200px;
    margin:30px 20px;
    padding:20px 10px;
    border:1px solid #000;
    text-align:justify
}
```

Với ví dụ này, chúng ta sẽ khái quát được mô hình hộp như sau:



Hình 17. Kết quả thực hiện

### 4.3 Margin & padding

Như tất cả những ai đã học qua MS Word đều biết là trong phần thiết lập Page Setup của Word cũng có một thiết lập margin để định lề cho trang in. Tương tự, thuộc tính margin trong CSS cũng được dùng để canh lề cho cả trang web hay một thành phần web này với các thành phần web khác hay với viền trang.

#### 4.3.1. Thuộc tính margin

Margin dùng để xác định khoảng cách giữa nó và đối tượng bao quanh nó. Có thể sử dụng 4 thuộc tính của margin bao gồm:

```
margin-left: length/percent/auto;  
margin-right: length/percent/auto;  
margin-top: length/percent/auto;  
margin-bottom: length/percent/auto;
```

Ví dụ sau cho biết cách canh lề cho một trang web.

```
body {  
    margin-top: 80px;  
    margin-bottom: 40px;  
    margin-left: 50px;  
    margin-right: 30px;  
    border: 1px dotted #FF0000}
```

Hoặc gọn hơn sẽ viết như sau:

```
body {  
    margin: 80px 30px 40px 50px;  
    border: 1px dotted #FF0000}
```

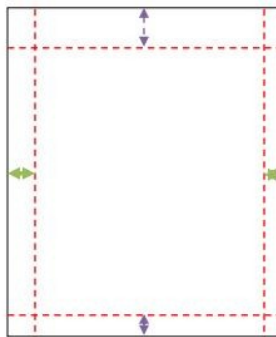
**Cú pháp:**

```
margin:<margin-top> | <margin-right> | <margin-bottom> |  
      <margin-left>
```

**Hoặc:**

```
margin:<value1>|< value2>
```

Với value1 là giá trị margin-top và margin-bottom và value2 là giá trị margin-left và margin-right. Kết quả của ví dụ trên sẽ được mô hình hóa như sau:



Hình 18. Mô hình hóa bằng thuộc tính margin

Ví dụ kế tiếp sẽ thể hiện rõ hơn về việc dùng margin để canh lề cho các đối tượng trong trang web. Hãy quan sát các đường viền và nhận xét.

```
body {  
    margin:80px 30px 40px 50px;  
    border:1px solid #FF0000  
}  
#box1 {  
    margin:50px 30px 20px 40px;  
    border:1px solid #00FF00  
}  
#box2 {  
    margin:50px 30px 20px 40px;  
    border:1px solid #0000FF  
}
```

#### 4.3.2. Thuộc tính padding

Thuộc tính padding có thể hiểu như là một thuộc tính đệm. Thuộc tính Padding không ảnh hưởng tới khoảng cách giữa các đối tượng như margin mà nó chỉ quy định khoảng cách giữa phần nội dung và viền của một đối tượng (xem lại ảnh minh họa về Model Box –).

**Cú pháp:**

<code>&lt;padding-top&gt;   &lt;padding-right&gt;   &lt;padding-bottom&gt;   &lt;padding-left&gt;</code>
--

#### 4.4 Border

Thuộc tính Border (đường viền) trong CSS cho phép người dùng định dạng đường viền cho các phần tử dạng khối (block) trong tài liệu HTML.

**Thuộc tính border-width:** được sử dụng để chỉ định độ dày của đường viền

**Cú pháp:**

```
border-width: value;
```

Trong đó giá trị value có thể là: Length, Thin, Medium, Thick

**Thuộc tính border-color:** được sử dụng để thiết lập màu sắc cho đường viền

**Cú pháp:**

```
border-color: value;
```

Trong đó giá trị value có thể là: tên màu, tên màu ở dạng Hexa, RGB color code, transparent - trong suốt, không màu.

**Thuộc tính border-style:** được sử dụng để thiết lập kiểu đường viền (đường thẳng, đường gạch nổi,...)

**Cú pháp:**

```
border-style: value;
```

Trong đó, value có thể là: dashed, dotted, double, groove, hidden, inset, none, outset, ridge, solid

## 4.5 Height & width

### 4.5.1. Thuộc tính width

Width là một thuộc tính CSS dùng để quy định chiều rộng cho một thành phần web. Ví dụ sau sẽ định chiều rộng cho thành phần p của một trang web.

Ví dụ: `p{ width:700px; }`

**Thuộc tính max-width:** là thuộc tính CSS dùng để quy định chiều rộng tối đa cho một thành phần web.

**Cú pháp:**

```
Thẻ_tag{ max-width: giá trị; }
```

Với giá trị như sau:

Thuộc tính	Giá trị	Ví dụ	Mô tả
max-width	<a href="#">Đơn vị</a>	max-width: 50px;	Chiều cao tính từ mép bên trái ngoài cùng của thành phần, <a href="#">đơn vị</a> có thể là px, em, %, ...
	none	max-width: none;	Không sử dụng chiều rộng lớn nhất, đây là dạng mặc định.
	inherit	max-width: inherit;	Xác định thừa hưởng thuộc tính từ thành phần cha (thành phần bao ngoài).

Ví dụ

```
<html>
  <body>
    <div>
      <p>Đoạn text này có chiều rộng lớn nhất
      là 120px.</p>
    </div>
  </body>
```

</html>

Giả sử bạn đầu CSS viết:

```
div { border: 1px solid red; }
```

Hiện thị trình duyệt khi chưa có thuộc tính max-width:

Đoạn text này có chiều rộng lớn nhất là 120px.

Thêm thuộc tính max-width vào CSS:

```
div {  
    border: 1px solid red;  
    max-width: 120px;  
}
```

Hiện thị trình duyệt khi đã thêm max-width vào CSS:

Đoạn text này có  
chiều rộng lớn  
nhất là 120px.

Khi đã sử dụng thuộc tính max-width thì chiều rộng của box sẽ không vượt quá giá trị của max-width.

**Thuộc tính min-width:** thiết lập chiều rộng tối thiểu (nhỏ nhất) cho thành phần.

**Cú pháp:**

```
Thẻ_tag { min-width: giá trị; }
```

Với giá trị như sau:

Thuộc tính	Giá trị	Ví dụ	Mô tả
min-width	<a href="#">đơn vị</a>	min-width: 200px;	Chiều rộng tối thiểu tính từ mép trên cùng của thành phần, <a href="#">đơn vị</a> có thể là px, em, %, ...
	inherit	min-width: inherit;	Xác định thừa hưởng thuộc tính từ thành phần cha (thành phần bao ngoài).

Ví dụ

```
<html>
```

```
<body>
```

```
<div>
```

```
<ul>
```

```
<li><a href="html4-xhtml.php">Tham khảo  
html4</a></li>
```

```
<li><a href="html5/">Tham khảo html5</a></li>
```

```
        <li><a href="cssSection/">Tham khảo
css2</a></li>
        <li><a href="cssSection/">Tham khảo
css3</a></li>
        <li><a href="tag/">Tham khảo jquery</a></li>
    </ul>
</div>
</body>
</html>
```

Giả sử ban đầu CSS viết:

```
div { border: 1px solid red; }
```

Hiện thị trình duyệt khi chưa có thuộc tính min-width:



Thêm thuộc tính min-width vào CSS:

```
div {
    border: 1px solid red;
    min-width: 180px;
}
```

Hiện thị trình duyệt khi đã thêm min-width vào CSS:



Khi nội dung dài thì chiều rộng của box được tăng thêm, nhưng khi nội dung ngắn thì giá trị chiều rộng nhỏ nhất sẽ bằng với giá trị min-width, chiều rộng lúc này không thể nhỏ hơn giá trị min-width.



#### 4.5.2. Thuộc tính height

Thuộc tính height thiết lập chiều cao cho thành phần. Chiều cao này không bao gồm: [border](#), [padding](#), [margin](#).

**Cú pháp:**

```
Thẻ_tag { height: giá trị; }
```

Với giá trị như sau:

Thuộc tính	Giá trị	Ví dụ	Mô tả
height	<a href="#">đơn vị</a>	height: 100px;	Định rõ đơn vị cho chiều cao.
	auto	height: auto;	Định chiều cao tự động, đây là dạng mặc định.
	inherit	height: inherit;	Xác định thừa hưởng thuộc tính từ thành phần cha (thành phần bao ngoài).

*Ví dụ:*

```
<html>
  <body>
    <p>HỌC WEB HAY</p>
  </body>
</html>
```

Hiện thị trình duyệt khi chưa có CSS:

HỌC WEB HAY

CSS viết:

```
p {
  background: #cccccc;
  height: 100px;
}
```

Hiện thị trình duyệt khi có CSS:

HỌC WEB HAY

**Thuộc tính max-height:** thiết lập chiều cao tối đa cho thành phần, về cách sử dụng tương tự như max-width tuy nhiên khi áp dụng sẽ tương tác với chiều cao

**Cú pháp:**

```
Thẻ_tag { max-height: giá trị; }
```

Với giá trị như sau:

Thuộc tính	Giá trị	Ví dụ	Mô tả
max-height	đơn vị	max-height: 50px;	Chiều cao tối đa tính từ mép trên cùng của thành phần, đơn vị có thể là px, em, %, ...
	none	max-height: none;	Không sử dụng chiều cao lớn nhất, đây là dạng mặc định.
	inherit	max-height: inherit;	Xác định thừa hưởng thuộc tính từ thành phần cha (thành phần bao ngoài).

**Thuộc tính min-height:** thiết lập chiều cao tối thiểu (nhỏ nhất) cho thành phần.

**Cú pháp:**

Thẻ\_tag { min-height: giá trị; }

Với giá trị như sau:

Thuộc tính	Giá trị	Ví dụ	Mô tả
min-height	đơn vị	min-height: 200px;	Chiều cao tối thiểu tính từ mép trên cùng của thành phần, <u>đơn vị</u> có thể là px, em, %, ...
	inherit	min-height: inherit;	Xác định thừa hưởng thuộc tính từ thành phần cha (thành phần bao ngoài).

## 4.6 Float & clear

### 4.6.1. Thuộc tính float

Thuộc tính float xác định có hay không một thành phần được float (trôi nổi).

**Cú pháp:**

Thẻ\_tag { float: giá trị; }

Với giá trị như sau:

Thuộc tính	Giá trị	Ví dụ	Mô tả
Float	left	float: left;	Thành phần được trôi nổi (float) qua bên trái ...

	right	float: right;	Thành phần được trôi nổi (float) qua bên phải.
	none	float: none;	Thành phần không được trôi nổi (float) qua bên phải hay trái, đây là dạng mặc định.
	inherit	float: inherit;	Xác định thừa hưởng thuộc tính từ thành phần cha (thành phần bao ngoài).

*Ví dụ:*

```
<html><body>
    <p>Float left</p>
    <p>Float left</p>
</body>
</html>
```

Hiển thị trình duyệt khi chưa có CSS:

Float left

Float left

CSS viết:

```
p {
    float: left;
}
```

Hiển thị trình duyệt khi có CSS:

Float left Float left

#### 4.6.2. Thuộc tính clear

Thuộc tính clear xác định 2 bên của phần tử (left, right), nơi mà phần tử [float](#) không được cho phép (ngăn cản thành phần không được float trái, phải hay cả hai).

**Cú pháp:**

Thẻ\_tag {clear: giá trị; }

Với giá trị như sau:

Thuộc tính	Giá trị	Ví dụ	Mô tả
clear	left	clear: left;	Bên trái của thành phần không được float.

	right	clear: right;	Bên phải của thành phần không được float.
	both	clear: both	Bên trái và phải của thành phần không được float.
	none	clear: none;	Đây là mặc định của thành phần clear, bên trái và phải của thành phần được float.
	inherit	clear: inherit;	Xác định thừa hưởng thuộc tính từ thành phần cha (thành phần bao ngoài).

*Ví dụ:*

```
<html>
  <body>
    <p class="exFloat">
      
    </p>
    <p>Dòng text này không sử dụng thuộc tính
    clear, nên bị ảnh hưởng float của thành phần p
    có class = exFloat</p>
    <p class="exClear">Dòng text này có sử dụng
    thuộc tính clear, nên không bị ảnh hưởng float
    của thành phần p class=exFloat</p>
  </body>
</html>
```

CSS viết:

Giả sử, ta có một thành phần dùng float: left; như sau

```
p.exFloat {float: left;}
```



Dòng text này không sử dụng thuộc tính clear, nên bị ảnh hưởng float của thành phần p có class = exFloat.

Dòng text này có sử dụng thuộc tính clear, nên không bị ảnh hưởng float của thành phần p class = exFloat


*Hình 19 Ví dụ về float trong CSS*

Thêm thuộc tính clear vào CSS:

```
p.exFloat {float: left;}
```

```
p.exClear {clear: left;}
```

Hiển thị trình duyệt khi dùng thuộc tính clear left:

	Dòng text này không sử dụng thuộc tính clear, nên bị ảnh hưởng float của thành phần p có class = exFloat
Dòng text này có sử dụng thuộc tính clear, nên không bị ảnh hưởng float của thành phần p class = exFloat	

Hình 20 Ví dụ về thuộc tính clear trong CSS

## 4.7 Position

Thuộc tính position trong CSS dùng để xác định vị trí hiển thị cho thẻ HTML và thường được dùng để xây dựng CSS cho menu đa cấp, tooltip hoặc một số chức năng khác. Position có tổng cộng 5 giá trị như bảng dưới đây:

Tên giá trị	Ý nghĩa
static	Dạng mặc định - sẽ hiển thị theo đúng thứ tự của nó (thường dùng để hủy các thuộc tính bên dưới).
relative	Định vị trí tương đối theo thẻ cha (thẻ khai báo relative) hoặc thẻ body nếu ko có khai báo.
absolute	Định vị trí tuyệt đối (vị trí bao ngoài), lúc này các thẻ HTML bên trong sẽ coi nó là thẻ cha.
fixed	Định vị trí tương đối cho cửa sổ Browser của trình duyệt (khi kéo scroll nó sẽ không bị ẩn đi).
inherit	Thừa hưởng các thuộc tính từ thành phần cha (thành phần bao ngoài nó).

### 4.7.1. Absolute position

Định vị tuyệt đối là sự định vị mà trong đó các thành phần được định vị không để lại bất cứ một khoảng trống nào trong tài liệu. Một thành phần được định vị tuyệt đối sẽ nhận giá trị position là absolute. Các đối tượng đã định vị tuyệt đối sẽ dùng kết hợp với các thuộc tính top, left, right, bottom để xác định tọa độ.

```
<div id="container">
  <div id="relative">
    
```

```

    </div>
</div>
Viết đoạn mã CSS:
div {
    box-sizing: border-box;
}
#container {
    width: 960px;
    margin: 0 auto;
    background-color: #e6e6e6;
    border: 1px solid #333;
    padding: 15px;
}
#relative{
    width: 100%;
    border: 1px solid #333;
    background-color: red;
    height: 300px;
    position: relative;
}
#absolute {
position: absolute;
right: 0;
bottom: 0;}

```



*Hình 21. Ví dụ về Absolute position*

#### *4.7.2. Relative position*

Sự định vị tương đối cho một thành phần là sự định vị được tính từ vị trí gốc trong tài liệu. Các thành phần đã được định vị tương đối sẽ để lại khoảng không trong tài liệu. Các thành phần được định vị tương đối sẽ nhận giá trị position là relative. Hãy làm lại ví

dụ trên nhưng thay absolute thành relative. Hãy ghi nhận lại vị trí 4 ảnh logo lúc áp dụng thuộc tính position là none, absolute và relative rồi rút ra nhận xét.

Ví dụ:

```

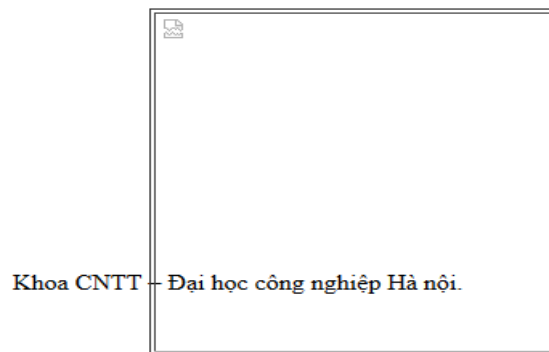
```

```
<p>Khoa CNTT - Đại học công nghiệp Hà nội.</p>
```

Viết đoạn mã CSS:

```
img {  
    position: relative;  
    top: 85px;  
    left: 85px;  
    border: 1px solid #333;  
    padding: 2px;  
}
```

Kết quả thể hiện:



Hình 22. Ví dụ về relative position