

Chương 4. CASCADING STYLE SHEET (CSS)

CSS phá vỡ giới hạn trong thiết kế Web, bởi chỉ cần một file CSS có thể cho phép các nhà phát triển Web quản lý định dạng và layout trên nhiều trang khác nhau. Với CSS ta có thể định nghĩa sẵn thuộc tính của một số thẻ HTML để sau đó dùng lại trên nhiều trang khác.

Contents

Chương 4. CASCADING STYLE SHEET (CSS).....	1
4.1 GIỚI THIỆU	2
4.1.1 CSS là gì?.....	2
4.1.2 Ưu điểm của CSS.....	2
4.2 CÚ PHÁP CSS	2
4.3 CÁC BỘ SELECTOR TRONG CSS	4
4.1.3 Selector đơn	4
4.4 CHÈN CSS VÀO TRANG WEB.....	5
4.4.1. Tạo 1 file CSS riêng (External Style Sheet)	6
4.5 Chèn trực tiếp câu lệnh CSS trong phần <Head> của tài liệu HTML (Internal Style Sheet) 6	7
4.6 Chèn trực tiếp CSS vào các thẻ của HTML (Inline Styles)	7
3.3.2. Sự ưu tiên	8
4.7 CÁC THUỘC TÍNH ĐỊNH DẠNG CƠ BẢN TRONG CSS.....	10
4.7.1 Nhóm thuộc tính định dạng nền (Background).....	10
4.7.2. Nhóm thuộc tính định dạng font	12
4.7.2.1 Nhóm thuộc tính text	15
4.7.2. Border, margin và padding	16
4.7.3. Định dạng liên kết (Links)	21
4.7.4. Định dạng danh sách (Lists)	22
4.7.5. Table	23
4.8. MỘT SỐ TÍNH NĂNG MỚI TRONG CSS3	25
4.8.1. Thumbnail Graphics	25
4.8.2. Làm việc với CSS3 Transitions	28

4.8.3. CSS3 Animation	29
4.9. KẾT HỢP THẺ div VÀ CSS ĐỂ TẠO BỐ CỤC TRANG WEB	29
4.8 MỘT SỐ VÍ DỤ	31
4.9.....	37
CÂU HỎI VÀ BÀI TẬP CHƯƠNG 4.....	38
TÀI LIỆU THAM KHẢO CHƯƠNG 4	42
1. Jon Duckett (2014). <i>HTML & CSS Design and Build Websites</i> . Wiley.	42
2. http://www.w3schools.com/css/default.asp	42

4.1 GIỚI THIỆU

4.1.1 CSS là gì?

CSS là chữ viết tắt của cụm từ tiếng Anh **Cascading Style Sheet** được tổ chức W3C giới thiệu vào năm 1996. Một file CSS có phần mở rộng là .css, trong file này chứa những câu lệnh CSS. Mỗi một lệnh của CSS sẽ định dạng một phần nhất định của tài liệu HTML như font chữ, đường viền, màu nền, căn chỉnh hình ảnh v.v..

Trước đây, khi chưa có CSS, những người thiết kế web phải trộn lẫn giữa các thành phần trình bày và nội dung với nhau. Nhưng với sự xuất hiện của CSS, người ta có thể tách rời hoàn toàn phần trình bày và nội dung. Điều này, giúp cho phần code của trang web gọn hơn và khi cần sửa thì dễ dàng hơn.

4.1.2 Ưu điểm của CSS

- Tách riêng phần định dạng ra khỏi nội dung trang web.
- CSS giúp người thiết kế kiểm soát toàn bộ giao diện nhanh nhất và hiệu quả nhất. CSS giúp ta tiết kiệm công sức rất nhiều trong việc thiết kế giao diện.

Ví dụ: Ta muốn font chữ của toàn bộ các trang web trong website là Arial mà không dùng CSS, thì sẽ phải làm đi làm lại phần định dạng Font chữ cho tất cả các file .html. Nhưng nếu sử dụng CSS, thì chỉ cần làm một lần và tất cả các trang khác sẽ tự động được thay đổi.

4.2 CÚ PHÁP CSS

Cú pháp:

Selector {Property: value;}

Trong đó:

+ **Selector:** Là các thẻ HTML, CLASS hoặc ID mà chúng ta sẽ áp dụng các thuộc tính trình bày.

+ **Property:** Chính là các thuộc tính quy định cách trình bày.

Ví dụ: background-color, font-family, color, padding, margin,...

+ **value:** Giá trị của thuộc tính trình bày. Nếu giá trị thuộc tính có nhiều từ thì nên đặt trong dấu nháy kép.

Chú ý: Trong trường hợp thẻ chọn có nhiều thuộc tính thì các thuộc tính sẽ được ngăn cách bởi dấu (;).

Ví dụ:

body {background:#FFF; color:#FF0000; font-size:14pt; font-family: "sans serif"}

Để dễ đọc hơn, nên viết mỗi thuộc tính CSS ở một dòng như sau:

```
body
{
    background:#FFF;
    color:#FF0000;
    font-size:14pt
    font-family: "sans serif"
}
```

Trong trường hợp nhiều thành phần có cùng một số thuộc tính, chúng ta có thể thực hiện thu gọn lại.

Ví dụ:

Đoạn code sau:

```
h1 {          color:#0000FF;
    text-transform:uppercase
}
h2 {
    color:#0000FF;
    text-transform:uppercase;
}
h3 {
    color:#0000FF;
    text-transform:uppercase;
}
```

Sẽ được thay thế bằng:

```
h1, h2, h3 { color:#0000FF;
```

```
text-transform:uppercase;
}
```

4.3 CÁC BỘ SELECTOR TRONG CSS

Có hai kiểu selector cơ bản: Selector đơn và Select ngữ cảnh

4.1.3 Selector đơn

Đây là những selector dễ sử dụng nhất. Có 3 kiểu selector đơn:

- Selector HTML (Bộ chọn phần tử)
- Selector Class (Bộ chọn lớp)
- Selector ID (Bộ chọn ID)

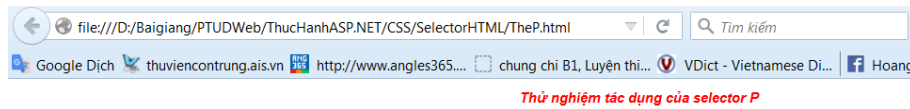
a. Selector HTML

Những selector này sử dụng tên của phần tử HTML và bỏ đi dấu <> . Ví dụ, thẻ <body> trong HTML trở thành *body* và nó được xem như là một selector. Ví dụ sau đây minh họa điều đó.

```
<HTML>
<HEAD>
<Title> Trang web thử nghiệm</Title>
<meta charset="UTF-8">
<STYLE TYPE="text/css">
body {
    color:#000;
    font:400 12px arial;
    text-align:left;
    background:white;
}
p {
    text-align:center;
    width:100%;
    color:limegreen;
    height:150px;
    text-indent:50px;
}
</STYLE>
</HEAD>
```

```
<BODY>
    <P>Thử nghiệm tác dụng của selector P </P>
</BODY>
</HTML>
```

Kết quả:



Hình 3-1. Sử dụng Selector P

b. Selector ID

Selector ID sử dụng thuộc tính ID của phần tử HTML. Selector ID được dùng để áp dụng một kiểu vào riêng một phần tử nào đó trên trang Web. Selector ID được bắt đầu bằng dấu thăng (#).

Ví dụ :

```
#banner
{
    text-align:center;
}
```

Trong ví dụ trên, dữ liệu text nằm trong phần tử HTML có id="banner" sẽ được căn giữa.

c. Selector Class

Dùng xác định style cho nhóm các đối tượng có cùng thuộc tính, do tính chất đó nó có thể được sử dụng cho nhiều thành phần HTML. **class selector** sử dụng thuộc tính class và được định nghĩa với kí hiệu "."

Ví dụ:

```
.center {text-align:center;}
```

Trong ví dụ trên, tất cả các phần tử HTML có class="center" sẽ được căn giữa.

Lưu ý: Tên của id và class giống như biến, không được bắt đầu bằng một số.

4.4 CHÈN CSS VÀO TRANG WEB

Có ba cách khác nhau để chèn CSS vào một tài liệu HTML:

- ✓ **Cách 1:** Định nghĩa CSS trong 1 file riêng, sau đó nhúng file này vào trang web khi cần.
- ✓ **Cách 2:** Chèn các lệnh CSS trong cặp thẻ <Head>....<Head> của tài liệu HTML
- ✓ **Cách 3:** Chèn trực tiếp các lệnh CSS vào thẻ HTML

Thông thường, trong thực tế người ta dùng cách đầu nhiều hơn vì nó phát huy được ưu điểm của nó. Tuy nhiên, chúng ta nên biết cả 2 cách sau vì đôi khi vẫn phải sử dụng.

4.4.1. Tạo 1 file CSS riêng (External Style Sheet)

Cách này khá lý tưởng khi một định dạng được áp dụng cho nhiều trang web. Với cách này ta có thể thay đổi định dạng của toàn bộ các trang web bằng cách chỉ cần thay đổi chỉ một file .css. Để mỗi trang web liên kết tới file .css ta sử dụng tag <link> trong cặp thẻ <Head>...</Head> của tài liệu HTML theo cú pháp sau:

```
<HTML>

<head>
  <link rel="stylesheet" type="text/css" href="mystyle.css">
</head>

<body>
  <!--Nội dung -->
</body>

<HTML>
```

Đây là cách làm được đa phần các nhà thiết kế web dùng. Nó đặc biệt hữu ích cho việc đồng bộ hay bảo trì một website lớn sử dụng cùng một kiểu mẫu.

Một số lưu ý:

- Một file .css có thể được tạo bằng bất kỳ một trình soạn thảo code HTML nào (chẳng hạn như Notepad, Notepad++,).
- Trong file .css không chứa thẻ HTML.
- File phải được lưu vào bộ nhớ ngoài với phần mở rộng là .css

Ví dụ: Nếu những định dạng của thẻ <hr>, thẻ <p> và thẻ <body> dưới đây áp dụng cho nhiều trang HTML thì ta sẽ tạo 1 file .css có nội dung như sau:

```
hr {color:sienna;}
p {margin-left:20px;}
body {background-image:url("images/anhnen.gif");}
```

4.5 Chèn trực tiếp câu lệnh CSS trong phần <Head> của tài liệu HTML (Internal Style Sheet)

Cách này dùng trong trường hợp những định dạng CSS chỉ dành riêng cho tài liệu HTML đó.

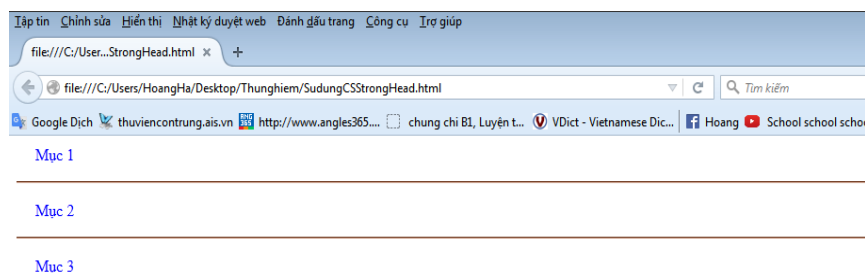
Chú ý: Đoạn mã CSS phải đặt trong thẻ <style> và đặt trong phần <head> của tài liệu HTML.

Ví dụ: Tạo 1 trang web có định dạng thẻ <hr> là màu nâu đỏ, đoạn văn nằm trong cặp thẻ <p>...</p> có màu xanh dương, cách mép trái 20px:

```
<html>

<head>
```

```
<style>
    hr {color:sienna;}
    p {margin-left:20px;color:Blue}
</style>
</head>
<Body>
    <p>Mục 1</p>
    <hr>
    <p>Mục 2</p>
    <hr>
    <p>Mục 3</p>
</Body>
</html>
```



Hình 3-2. Sử dụng CSS trong phần Head

4.6 Chèn trực tiếp CSS vào các thẻ của HTML (Inline Styles)

Cách này còn gọi là CSS nội tuyến, được sử dụng trong trường hợp một thẻ HTML nào đó cần có định dạng riêng và được áp dụng cho chính thẻ HTML đó. Cách này, mã CSS chèn trực tiếp vào thẻ HTML với cú pháp <Tên thẻ HTML style=Property: value;> và chỉ có tác dụng lên thẻ đó. Vì vậy, nó không phát huy được những ưu điểm của css nên ít khi dùng.

Ví dụ: Tạo 1 trang web có màu nền hồng, đoạn văn bản 1 chữ đậm, màu xanh lá cây, đoạn văn bản 2 chữ thường, màu xanh dương, đoạn văn bản 3 chữ thường, màu đen:

```
<html>
<head>
    <title>Ví dụ</title>
</head>
```

```
<body style="background-color:Pink">

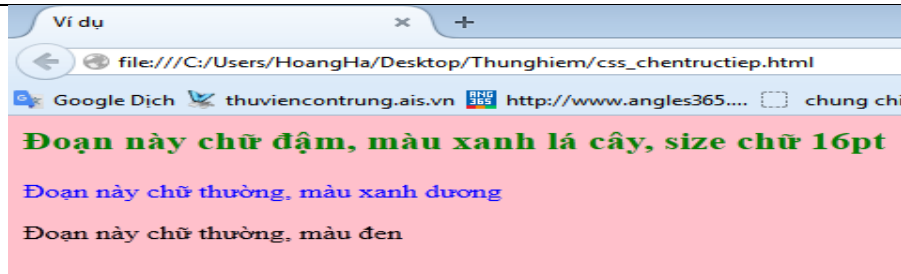
  <p style="color:green; font-size:16pt; font-weight:bold"> Đoạn này chữ đậm,
  màu xanh lá cây, size chữ 16pt </p>

  <p style="color:Blue"> Đoạn này chữ thường, màu xanh dương</p>

  <p> Đoạn này chữ thường, màu đen</p>

</body>

</html>
```



Hình 3-4. Sử dụng CSS trực tiếp vào các thẻ của HTML

3.3.2. Sự ưu tiên

Trước khi thực thi CSS cho một trang web, trình duyệt sẽ đọc toàn bộ CSS mà trang web có thể được áp dụng, bao gồm: CSS mặc định của trình duyệt, file CSS bên ngoài liên kết vào trang web, CSS nhúng trong cặp thẻ `<style>...</style>` và các CSS nội tuyến. Sau đó, trình duyệt sẽ tổng hợp toàn bộ CSS này vào một CSS ảo, và nếu có các thuộc tính CSS giống nhau thì thuộc tính CSS nào nằm sau sẽ được ưu tiên sử dụng. Theo nguyên tắc đó trình duyệt sẽ ưu tiên cho các CSS nội tuyến → CSS bên trong cặp thẻ `<style>...</style>` → CSS bên ngoài → CSS mặc định của trình duyệt.

Ví dụ:

Trong một trang web có liên kết tới file **style.css** có nội dung như sau:

```
p {
    color:#333;
    text-align:left;
    width:500px;
}
```

Trong thẻ `<style>` nằm trong cặp thẻ `<head>` cũng có một đoạn CSS như sau:

```
p {
    background-color:#FF00FF;
    text-align:right;
    width:100%;
}
```



```
height:150px;
}
```

Trong phần nội dung trang web đó thẻ <p> cũng có sử dụng CSS nội tuyến như sau:

```
<p style= "height:200px; text-align:center;color:#0F0">
```

Khi thực thi CSS trình duyệt sẽ đọc hết tất cả các nguồn chứa style rồi sẽ tổng hợp lại vào một CSS ảo và nếu có sự trùng lặp các thuộc tính CSS thì nó sẽ lấy thuộc tính CSS có mức ưu tiên cao hơn. Như ví dụ trên, chúng ta sẽ thấy CSS cuối cùng mà phần tử **p** nhận được là:

```
p {
    background-color:#FF00FF;
    width:100%;height:200px;
    text-align:center;
    color:#0F0;
}
```

Vậy có cách nào để thay đổi độ ưu tiên cho một thuộc tính nào đó? Trong CSS đã có sẵn một thuộc tính giúp chúng ta thực hiện điều này, đó chính là thuộc tính **!important**. Chỉ cần đặt thuộc tính này sau một thuộc tính nào đó theo cú pháp **selector { property:value !important }** thì trình duyệt sẽ hiểu đây là một thuộc tính được ưu tiên. Bây giờ, chúng ta cùng xét lại ví dụ trên nhưng có đặt thêm một số thuộc tính **!important** vào xem kết quả như thế nào nhé.

```
p {
    width:500px;
    text-align:left !important;
    color:#333 !important
}
p {
    background-color:#FF00FF;
    width:100%;
    height:150px !important;
    text-align:right;
}
<p style="text-align:center; height:200px; border:1px solid #FF0000; color:#0F0">
```

Phần CSS sẽ tác động lên thuộc tính p là:

```
p {
```

```
background-color:#FF0000;  
width:100%;  
height:150px !important;  
text-align:left !important;  
border:1px solid #FF0000;  
color:#333 !important  
}
```

Lưu ý: Cùng một thuộc tính cho một selector, nếu cả hai thuộc tính đều đặt !important thì hệ thống sẽ chọn cái sau.

4.7 CÁC THUỘC TÍNH ĐỊNH DẠNG CƠ BẢN TRONG CSS

4.7.1 Nhóm thuộc tính định dạng nền (Background)

Thuộc tính **background** là một trong các thuộc tính khá phổ biến của CSS.

Có 5 tính chất chính của background cần biết đó là:

- **background-color** : Đặt thuộc tính màu nền
- **background-image** : Dùng ảnh làm nền cho các thành phần.
- **background-position** : Vị trí của ảnh
- **background-repeat** : Tùy chỉnh ảnh nền có lặp hay không, lặp kiểu nào...
- **background-attachment** : Xác định thành phần nền được cố định hoặc cuộn so với trang. Được sử dụng kèm với giá trị background-image

a. Thuộc tính background-color

Thuộc tính background-color dùng để định dạng màu nền cho một thành phần được chỉ định trên trang web.

Ví dụ sau đây sẽ chỉ cho chúng ta biết cách sử dụng thuộc tính background-color để định màu nền cho cả trang web, màu nền cho các thành phần h1, h2 lần lượt là xanh lơ, đỏ và cam.

```
body {  
    background-color:cyan;  
}  
h1 {  
    background-color:red;  
}  
h2 {  
    background-color:orange;  
}
```

b. Thuộc tính background-image

Việc sử dụng ảnh nền giúp cho trang web trông sinh động và bắt mắt hơn. Để chèn ảnh nền vào một thành phần trên trang web chúng ta sử dụng thuộc tính background-image.

Ví dụ: Đặt ảnh có tên anhnen.gif làm màu nền cho trang web.

```
body {background-image:url('anhnen.gif');}
```

c. Thuộc tính background-position

Theo mặc định, ảnh nền khi được chèn sẽ nằm ở góc trên, bên trái màn hình. Tuy nhiên với thuộc tính background-position ta sẽ có thể đặt ảnh nền ở bất cứ vị trí nào (trong không gian của thành phần mà nó làm nền). Background-position sẽ dùng một cặp 2 giá trị để biểu diễn tọa độ đặt ảnh nền. Có khá nhiều kiểu giá trị cho thuộc tính position. Như đơn vị chính xác như centimeters, pixels, inches,... hay các đơn vị qui đổi như %, hoặc các vị trí đặt biệt như top, bottom, left, right

Bảng 3-1. Thuộc tính Background-position

Giá trị	Ý nghĩa
Background-position:5cm 2cm	Ảnh được định vị 5cm từ trái qua và 2cm từ trên xuống.
Background-position:20% 30%	Ảnh được định vị 20% từ trái qua và 30% từ trên xuống.
Background-position:bottom left	Ảnh được định vị ở góc trái phía dưới

d. Thuộc tính background-repeat

Nếu sử dụng một ảnh có kích thước quá nhỏ để làm nền cho một đối tượng lớn hơn thì theo mặc định, trình duyệt sẽ lặp lại ảnh nền để phủ kín không gian còn thừa. Thuộc tính background-repeat cung cấp cho chúng ta các điều khiển giúp kiểm soát tình trạng lặp lại của ảnh nền. Thuộc tính này có 4 giá trị:

- repeat-x: Chỉ lặp lại ảnh theo phương ngang.
- repeat-y: Chỉ lặp lại ảnh theo phương dọc.
- repeat: Lặp lại ảnh theo cả 2 phương, đây là giá trị mặc định.
- no-repeat: Không lặp lại ảnh.

Bây giờ, chúng ta hãy thêm thuộc tính background-repeat này vào ví dụ trên và xem kết quả.

```
body {  
    background-  
image:url(logo.png);  
    background-repeat:no-repeat;  
}
```

e. Thuộc tính background-attachment

- Background-attachment là một thuộc tính xác định tính cố định ảnh nền so với nội dung trang web. Thuộc tính này có 2 giá trị:
- scroll: Ảnh nền sẽ cuộn cùng nội dung trang web, đây là giá trị mặc định.
- fixed: Cố định ảnh nền so với nội dung trang web. Khi áp dụng giá trị này, ảnh nền sẽ đứng yên khi bạn đang cuộn trang web.

Ví dụ:

```
body{
background-image:url('anhnen.gif');
background-repeat:no-repeat;
background-attachment:fixed;
background-position:center;
}
```

f. Thuộc tính background rút gọn

Khi sử dụng quá nhiều thuộc tính CSS sẽ gây mất thời gian cho người thiết kế và tốn nhiều dung lượng ổ cứng, cho nên CSS đưa ra một cấu trúc rút gọn cho các thuộc tính cùng nhóm.

Ví dụ: Đoạn mã CSS sau:

```
background-color:transparent;
background-image: url(File_anhnen);
background-repeat: no-repeat;
background-attachment: fixed;
background-position: right bottom;
```

Chúng ta có thể nhóm thành một dòng ngắn gọn sau:

```
background:transparent url(anhnen) no-repeat fixed right bottom;
```

Từ ví dụ trên chúng ta có thể khái quát cấu trúc rút gọn cho nhóm:

```
background:<background-color> | <background-image> | <background-repeat>  
|<background-attachment> | <background-position>
```

Theo mặc định thì các thuộc tính không được đề cập sẽ nhận các giá trị mặc định.

4.7.2. Nhóm thuộc tính định dạng font

Nhóm thuộc tính font thuộc nhóm thuộc tính định dạng văn bản bao gồm 5 thuộc tính: font-family, font-style, font-variant, font-weight, font-size cho phép xác định kiểu chữ, độ cao chữ và kích thước chữ.

a. Thuộc tính font family

Thuộc tính font-family được dùng để định nghĩa Font chữ cho các thành phần. Nó định nghĩa một danh sách ưu tiên các font sẽ được dùng để hiển thị một thành phần trang web. Theo đó, thì font đầu tiên được liệt kê trong danh sách sẽ được dùng để hiển thị cho

trang web. Nếu như trên máy tính truy cập chưa cài đặt font này thì font thứ hai trong danh sách sẽ được ưu tiên...cho đến khi có một font phù hợp.

Có hai loại tên font được dùng để chỉ định trong font-family: family-names và generic families.

- Family-names: Tên cụ thể của một font. Ví dụ: Arial, Verdana, Tohama,...
- Generic families: Tên của một họ gồm nhiều font. Ví dụ: sans-serif, serif,...

Cú pháp:

```
tag {  
    font-family: kiểu chữ;  
}
```

Ví dụ: Viết CSS để quy định font chữ dùng cho cả trang web là Times New Roman, Tohama, sans-serif và font chữ dùng để hiển thị các tiêu đề h1, h2, h3 sẽ là Arial, Verdana và các font họ serif.

Body {font-family: “Times New Roman”, Tohama, sans-serif}

h1, h2, h3 {font-family: arial, verdana, sans-serif; }

b. Thuộc tính font-style

Thuộc tính font-style định nghĩa việc áp dụng các kiểu in thường (normal), in nghiêng (italic) hay xiên (oblique) lên các thành phần trang web.

```
tag {  
    font-style: italic/oblique/normal;  
}
```

Ví dụ: Áp dụng kiểu in nghiêng cho thành phần h1:

```
h1 {  
    font-style:italic;  
}
```

c. Thuộc tính font-weight

Thuộc tính font-weight mô tả cách thức thể hiện của font chữ là ở dạng bình thường (normal) hay in đậm (bold).

Cú pháp:

```
tag {  
    font-weight: normal/bold;  
}
```

Ví dụ: Đặt thuộc tính để in đậm phần tử p:

```
p {
```

```
font-weight:bold;
}
```

d. Thuộc tính font-size

Thuộc tính font-size được dùng để định dạng cỡ chữ cho thành phần.

Cú pháp:

```
tag {
    font-size: giá trị;
}
```

Giá trị có thể được tính theo các loại đơn vị khác nhau (thường dùng nhất là dạng điểm ảnh - px, dạng phần trăm - %, hoặc dạng em).

Ví dụ: Định dạng trang web có kích cỡ font là 20px, h1 là 3em = 3 x 20 = 60px, h2 là 2em = 40px.

```
body {
    font-size:20px;
}
h1 {
    font-size:3em;
}
h2 {
    font-size:2em;
}
```

Cấu trúc rút gọn cho các thuộc tính nhóm font:

Font:<font-style> | <font-weight> | <font-size> | <font-family>

Ví dụ:

```
p{
    font-style: italic;
    font-size: 12px;
    font-variant:small-caps;
    font-weight: bold;
    font-family: Arial, Verdana, Sans-serif;
}
```

Dạng viết giản lược:

```
p {font: italic bold 12px Arial,Verdana,Sans-serif;}
```

4.7.2.1 Nhóm thuộc tính text

a. Thuộc tính định dạng màu cho đoạn văn bản

Để đặt màu cho một đoạn văn bản chúng ta có thể dùng thuộc tính color.

Cú pháp:

```
tag {  
    color: giá trị màu;  
}
```

Ví dụ sau chúng ta sẽ viết CSS để định màu chữ chung cho trang web là đỏ, tiêu đề h1 màu xanh da trời, tiêu đề h2 màu xanh lá cây:

```
body {color:Red;}  
h1 {color:#0000FF;}  
h2 {color:#00FF00;}
```

b. Thuộc tính căn lề cho đoạn văn bản

Dùng để căn lề văn bản theo chiều ngang. Ta có thể căn lề trái, phải, giữa hoặc căn đều 2 bên cho đoạn văn bản bởi thuộc tính **text-align**:

```
tag {  
    text-align:right/Left/center/justify;  
}
```

c. Thuộc tính text-decoration

Thuộc tính text-decoration giúp ta thêm các hiệu ứng gạch chân (underline), gạch ngang (line-through), gạch đầu (overline) và một hiệu ứng đặc biệt là văn bản nhấp nháy (blink).

Cú pháp:

```
tag {  
    text-decoration: underline/ line-through/ overline;  
}
```

Ví dụ:

```
h3 {text-decoration: underline; /* chữ gạch chân cho h3*/}  
h2 {text-decoration: line-through; /* Gạch ngang cho h2 */}  
h1 {text-decoration: overline; /* kẻ trên cho h1 */}
```

d. Thuộc tính khoảng cách giữa các dòng

Thuộc tính line-height được dùng để căn chỉnh khoảng cách giữa các dòng trong một đoạn văn bản.

Cú pháp:

```
tag {  
    line-height: giá trị khoảng cách;  
}
```

Ví dụ: Muốn căn chỉnh khoảng cách giữa các dòng trong thành phần <p> là 15px ta có thể viết trong css như sau:

```
p {  
    line-height: 15px;  
}
```

e. Thuộc tính thụt vào đầu dòng

Thuộc tính *text-indent* cung cấp khả năng tạo ra khoảng thụt đầu dòng cho dòng đầu tiên trong đoạn văn bản. Giá trị thuộc tính này là các đơn vị đo cơ bản dùng trong CSS.

Cú pháp:

```
tag {  
    Text-indent: vị trí;  
}
```

Ví dụ: Định dạng thụt đầu dòng một khoảng 30px cho dòng văn bản đầu tiên trong mỗi đoạn văn bản chứa trong cặp thẻ <p>

```
p {  
    text-indent: 30px;  
}
```

f. Text-transform

Text-transform là thuộc tính qui định chế độ in hoa hay in thường của văn bản mà không phụ thuộc vào văn bản gốc trên HTML. Thuộc tính này có tất cả 4 giá trị: uppercase (in hoa), lowercase (in thường), capitalize (in hoa ở ký tự đầu tiên trong mỗi từ) và none (không áp dụng hiệu ứng – mặc định).

Ví dụ dưới đây ta sẽ định dạng cho thành phần h1 là chữ in hoa, h2 là in hoa đầu mỗi ký tự:

```
h1 { text-transform: uppercase; }  
h2 { text-transform: capitalize; }
```

4.7.2. Border, margin và padding

Đây là những thành phần quan trọng trong một trang web. Nó thường được dùng trong trang trí, đóng khung cho một đối tượng cần nhấn mạnh, phân cách các đối tượng giúp trang web trông dễ nhìn hơn.

a. Thuộc tính border

Thuộc tính **border** trong CSS cho phép ta xác định kiểu, chiều rộng và màu sắc của đường viền bao quanh một phần tử.

border



Thuộc tính **border** trong CSS cho phép ta xác định kiểu, chiều rộng và màu sắc của đường viền bao quanh một phần tử.

Hình 3-4. Mô hình thuộc tính border trong CSS

Trong thuộc tính đường viền (border) chúng ta có 3 giá trị cơ bản đó là:

1. border-color:

2. border-width:

4. border-style:

- ❖ *Thuộc tính xác định kiểu của border:* Để xác định kiểu của đường viền ta sử dụng thuộc tính **border-style**. Chúng ta có thể gán cho thuộc tính này 9 giá trị khác nhau tương ứng với 9 kiểu đường viền khác nhau. Các giá trị thuộc tính đó là: dotted, dashed, solid, double, double, groove, ridge, inset, outset, none, hidden.

Code	Kết quả
<pre><html> <head> <style> p.dotted {border-style: dotted;} p.dashed {border-style: dashed;} p.solid {border-style: solid;} p.double {border-style: double;} p.groove {border-style: groove;} p.ridge {border-style: ridge;} p.inset {border-style: inset;} p.outset {border-style: outset;} p.none {border-style: none;} p.hidden {border-style: hidden;} p.mix {border-style: dotted dashed solid double;} </style> </head> <body> <h2>Các giá trị thuộc tính của border- styler</h2></pre>	<p>Các giá trị thuộc tính của border-styler</p> <p>Thuộc tính này xác định kiểu border được hiển thị</p> <p>A dotted border.</p> <p>A dashed border.</p> <p>A solid border.</p> <p>A double border.</p> <p>A groove border.</p> <p>A ridge border.</p> <p>An inset border.</p> <p>An outset border.</p> <p>Không có border.</p> <p>Ẩn border.</p> <p>trộn border.</p>

<pre><p>Thuộc tính này xác định kiểu border được hiển thị</p> <p class="dotted">A dotted border.</p> <p class="dashed">A dashed border.</p> <p class="solid">A solid border.</p> <p class="double">A double border.</p> <p class="groove">A groove border.</p> <p class="ridge">A ridge border.</p> <p class="inset">An inset border.</p> <p class="outset">An outset border.</p> <p class="none">Không có border.</p> <p class="hidden">Ẩn border.</p> <p class="mix">trộn border.</p> </body> </html></pre>	
--	--

Hình 3-5. Sử dụng thuộc tính kiểu của border trong CSS

❖ *Thuộc tính màu của đường viền*

Để đặt màu cho đường viền chúng ta sẽ đặt thông số màu cho thuộc tính **border-color**. Giá trị màu có thể được đặt bởi 4 màu cơ bản:

name - specify a color name, like "red"

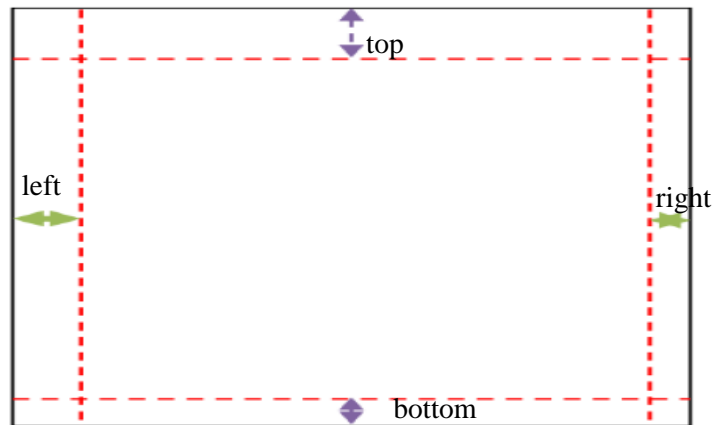
- Tên màu bằng tiếng Anh, ví dụ “red”, “green”,...
- RGB – xác định giá trị RGB, ví dụ "rgb(255,0,0)"
- Hex – xác định giá trị hex, ví dụ "#ff0000"
- transparent

❖ *Thuộc tính chiều rộng của đường viền*

Nếu muốn đặt chiều rộng của đường viền chúng ta sẽ đặt giá trị cho thuộc tính **border-width**. Các thuộc tính của **border-width** có thể nhận như sau: *thin, medium, thick, length*

a. *Thuộc tính margin*

Margin là thuộc tính canh lề có vai trò tương tự như margin trong Microsoft Word, dùng để canh chỉnh lề cho cả trang web, cho một thành phần nào đó đối với các thành phần khác hoặc canh lề đối với viền trang. Nó có thể dùng giá trị âm để lồng nội dung vào với nhau. Ứng với 4 phía của một phần tử chúng ta có 4 thuộc tính tương ứng: top, right, bottom, left:



Hình 3-6. Mô hình thuộc tính margin trong CSS

Cú pháp:

```
tag {  
    margin-top: giá trị;  
    margin-right: giá trị;  
    margin-bottom: giá trị;  
    margin-left: giá trị;  
}
```

Tất cả các thuộc tính của **margin** có thể có các giá trị sau:

- *auto* – trình duyệt tự tính các giá trị thuộc tính cho margin
- *length* – xác định khoảng cách margin (có thể sử dụng các đơn vị đo px, pt, cm).
- *%* - xác định khoảng cách margin dạng % chiều rộng của phần tử .

Để cho gọn, chúng ta cũng có thể viết thuộc tính **margin** theo nguyên tắc:

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

Thuộc tính **margin** có thể nhận từ 1 đến 4 giá trị:

- Nếu margin có 4 giá trị:

margin: 10px 15px 20px 25px;

Tương ứng với


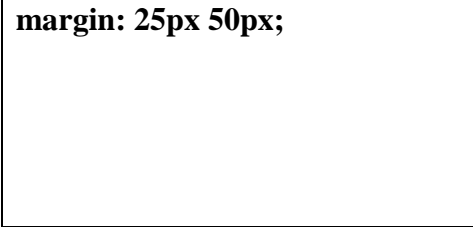
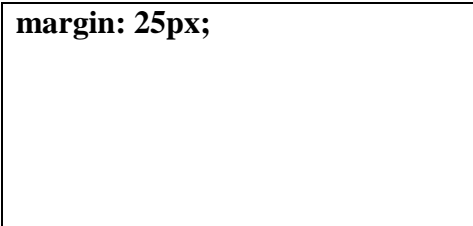
margin-top: 10px;
margin-right: 15px;
margin-bottom: 20px;
margin-left: 25px;

- Nếu margin có 3 giá trị:

margin: 25px 50px 35px;

Tương ứng với

margin-top: 25px;

	<div>margin-right: 50px; margin-left: 50px; margin-bottom: 35px;</div>
- Nếu margin có 2 giá trị:	
<div>margin: 25px 50px;</div> 	<div>Tương ứng với</div> <div>margin-top: 25px; margin-bottom: 25px; margin-left: 50px; margin-right: 50px;</div>
- Nếu margin có 1 giá trị:	
<div>margin: 25px;</div> 	<div>Tương ứng với</div> <div>margin-top: 25px; margin-right: 25px; margin-left: 25px; margin-bottom: 25px;</div>

b. Thuộc tính padding

padding định nghĩa khoảng trống giữa mép của các phần tử tới các phần tử con hoặc nội dung bên trong. Padding quy định khoảng cách giữa phần nội dung và viền của một đối tượng. Chúng ta không thể gán giá trị âm cho thuộc tính này. Tương ứng với 4 phía của phần tử chúng ta có 4 thuộc tính **padding** tương ứng.

Tất cả các thuộc tính của **margin** có thể có các giá trị sau:

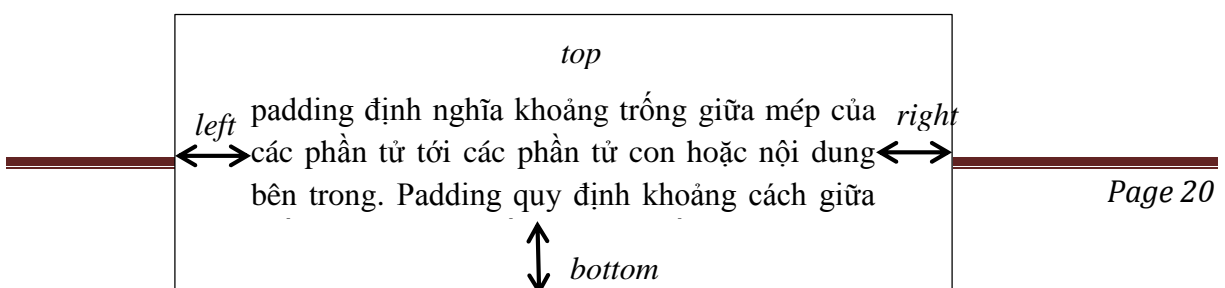
- auto – trình duyệt tự tính các giá trị thuộc tính cho margin
- length – xác định khoảng cách margin (có thể sử dụng các đơn vị đo px, pt, cm).
- % - xác định khoảng cách margin dạng % chiều rộng của phần tử .

Cú pháp:

```
tag {  
    padding-top: giá trị;  
    padding-right: giá trị;  
    padding-bottom: giá trị;  
    padding-left: giá trị;  
}
```

Để cho gọn, chúng ta cũng có thể viết thuộc tính **margin** theo nguyên tắc:

padding:<padding-top>|<padding-right>|<padding-bottom>|<padding-left>





Hình 3-7. Mô hình thuộc tính padding trong CSS

Cũng giống như margin, thuộc tính padding có thể nhận từ 1 đến 4 giá trị:

- Nếu padding có 4 giá trị:

padding: 10px 15px 20px 25px;	Tương ứng với	padding-top: 10px; padding-right: 15px; padding-bottom: 20px; padding-left: 25px;
--------------------------------------	----------------------	--

- Nếu padding có 3 giá trị:

padding: 25px 50px 35px;	Tương ứng với	padding-top: 25px; padding-right: 50px; padding-left: 50px; padding-bottom: 35px;
---------------------------------	----------------------	--

- Nếu padding có 2 giá trị:

padding: 25px 50px;	Tương ứng với	padding-top: 25px; padding-bottom: 25px; padding-left: 50px; padding-right: 50px;
----------------------------	----------------------	--

- Nếu padding có 1 giá trị:

padding: 25px;	Tương ứng với	padding-top: 25px; padding-right: 25px; padding-left: 25px; padding-bottom: 25px;
-----------------------	----------------------	--

4.7.3. . Định dạng liên kết (Links)

Một thành phần rất quan trọng trong mọi website chính là liên kết. Cũng như một đối tượng văn bản thông thường, chúng ta hoàn toàn có thể áp dụng các thuộc tính định dạng như 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 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).

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: Ví dụ này chúng ta 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  
}
```

4.7.4. Định dạng danh sách (Lists)

CSS có thể định dạng cho thẻ list trong HTML. Nó cho phép đặt kiểu danh sách. Danh sách trong HTML có hai loại:

- Danh sách không thứ tự
- Danh sách có thứ tự

Có thể xác định loại danh sách với thuộc tính list-style-type

a. Danh sách không thứ tự

```
ul {list-style-type: giá trị;}
```

b. Danh sách có thứ tự

```
ol {list-style-type: Giá trị;}
```

Dưới đây là một số giá trị của thuộc tính list-style-type thường được sử dụng:

Bảng 4.1. Các giá trị của thuộc tính list-style-type thường được sử dụng

Giá trị	Mô tả
None	Không hiển thị thứ tự
Disc	Chấm vuông

Circle	Chấm tròn trắng
Square	Chấm tròn đen
Decimal	Kiểu số
Lower-alpha	Kiểu Alphabet ở dạng in thường (a, b, c, d...)
Upper-alpha	Kiểu Alphabet ở dạng in hoa (A, B, C, D...)
Lower-roman	Kiểu La Mã ở dạng in thường (i, ii, iii, iv...)
upper-roman	Kiểu La Mã ở dạng in hoa (I, II, III, IV...)

c. Sử dụng ảnh cho thành phần danh sách

Có thể dùng một bộ ảnh như bộ đánh dấu thành phần danh sách marker

Cú pháp:

```
ul
{
    List-style-image:url('arrow.gif');
}
```

4.7.5. Table

Thuộc tính CSS table cho phép thiết lập các thuộc tính cho bảng.

a. Đường viền bảng

Để xác định đường viền bảng trong CSS ta sử dụng thuộc tính border.

Ví dụ:

```
table, th, td
{
    border: 1px solid black;
}
```

Chú ý: Để hiển thị một đường viền duy nhất cho bảng, sử dụng thuộc tính border-collapse.

```
table
{
    border-collapse:collapse;
}

table, th, td
{
    border: 1px solid black;
}
```

b. Kích thước bảng

Chiều rộng và chiều cao của bảng được xác định bởi các thuộc tính **width** và **height**. Ví dụ dưới đây đặt chiều rộng của bảng là 100% và chiều cao của phần tử th là 80px.

```
table
{
    width:100%;
}
th
{
    height:80px;
}
```

c. Căn chỉnh văn bản trong bảng

Các văn bản trong một bảng được định dạng bằng các thuộc tính text-align và vertical-align. Thuộc tính text-align thiết lập định dạng ngang như trái(left), phải(right) hoặc trung tâm(center):

Ví dụ: Định dạng dữ liệu trong 1 ô của bảng là căn phải

```
td
{
    text-align:right;
}
```

Thuộc tính vertical-align thiết lập định dạng theo chiều dọc với các giá trị: top (trên), middle (giữa), bottom (dưới).

Ví dụ: Định dạng dữ liệu trong 1 ô của bảng nằm chính giữa của ô theo chiều dọc:

```
td
{
    vertical-align:bottom;
}
```

d. Thiết lập không gian giữa các đường viền trong bảng

Để kiểm soát không gian giữa đường viền và nội dung trong một bảng sử dụng thuộc tính padding của phần tử td và th.

Ví dụ: Đặt không gian giữa đường viền và nội dung trong một ô của bảng là 15px:

```
td
{
    padding:15px;
}
```

e. Màu sắc của bảng

Ví dụ dưới đây xác định màu sắc của đường viền, màu sắc văn bản và nền của phần tử th, td:

```
table, td, th
{
```



```
border:1px solid green;
}
th
{
background-color:green;
color:white;
}
```

4.8. MỘT SỐ TÍNH NĂNG MỚI TRONG CSS3

4.8.1. Thumbnail Graphics

Tốc độ tải trang của một trang Web được giảm bớt nếu đồ họa độ phân giải cao được sử dụng.

Đồ họa độ phân giải cao được yêu cầu để nâng cao hiệu quả của các trang web và có thể không thể tránh được. Do đó, để tránh vấn đề này, hình thu nhỏ được sử dụng.

Một hình ảnh thu nhỏ là một hình ảnh nhỏ, hay một phần của một hình ảnh lớn hơn. Nhấp vào hình ảnh thu nhỏ sẽ liên kết đến hình ảnh ban đầu lớn hơn, có thể xem và tải về. Ngay cả một hiệu ứng di chuột qua hình ảnh thu nhỏ cũng có thể đưa ra hình ảnh ban đầu thông qua CSS và JavaScript.

Ví dụ : Mô tả cách sử dụng Thumbnail

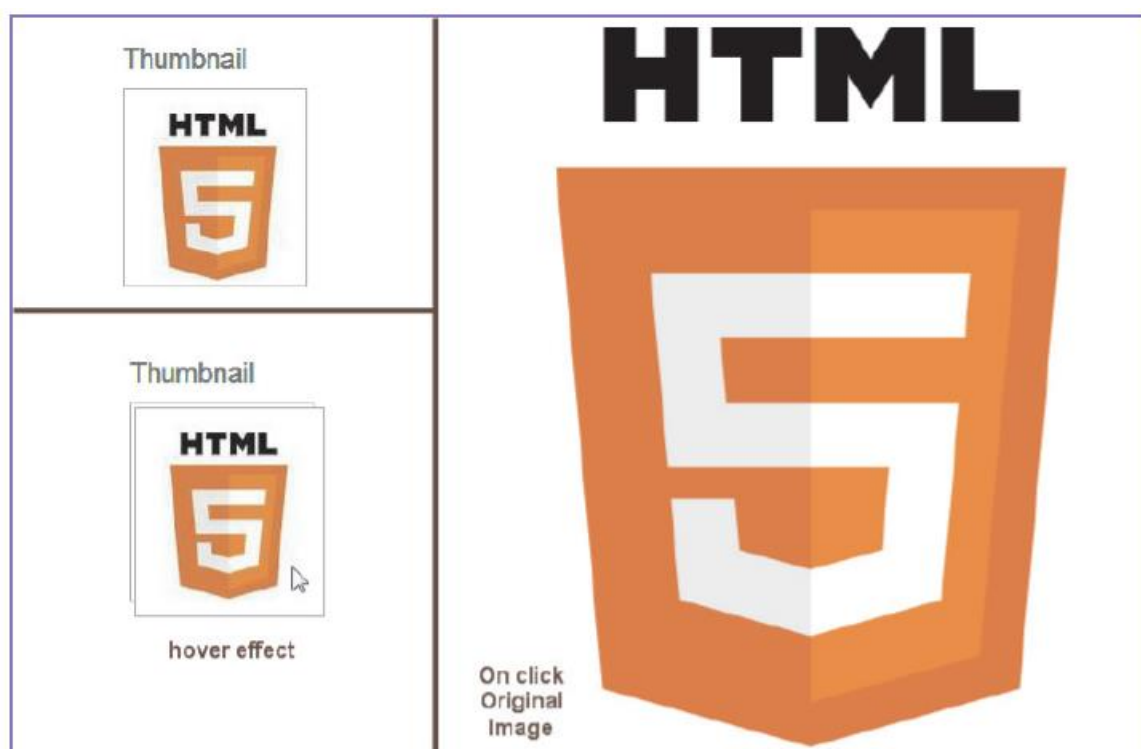
```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Thumbnail</title>
<style>
/* general */
body {
margin:0;

padding:40px 80px;
background:#fff;
font:70% Arial, Helvetica, sans-serif;
color:#555;
line-height:100%;
}
h1, h2{
font-size:180%;
font-weight:normal;
color:#555;
}
p{
margin:1em 0;
}
p.text{
width:500px;
}
a{
color:#f20;
text-decoration:none;
}
a:hover{
color:#000;
```

```
ul#thumbs li{
float:left;
margin-right:0px;
border:1px solid #999;
padding:2px;
}
ul#thumbs a{
display:block;
float:left;
width:125px;
height:135px;
line-height:50px;
overflow:hidden;
position:relative;
z-index:1;
}
ul#thumbs a img{
float:left;
position:absolute;
top:0px;
left:0px;
}
/* mouse over */
ul#thumbs a:hover{
overflow:visible;
z-index:1000;
border:none;
}
ul#thumbs a:hover img{
border:1px solid #999;
background:#fff;
padding:2px;
}
/* // mouse over */

/* clearing floats */
ul#thumbs:after, li#thumbs:after{
content:".";
display:block;
height:0;
clear:both;
visibility:hidden;
```

Kết quả:



4.8.2. Làm việc với CSS3 Transitions

Tính tương tác là một trong những khía cạnh quan trọng của hoạt hình. Trước đó, một sự kết hợp của HTML, CSS, và JavaScript được sử dụng để làm sinh động các đối tượng trên Web. Trong năm 2007, Apple đã giới thiệu chuyển động CSS, mà sau này đã trở thành một tính năng độc quyền của Safari gọi là CSS Animation. Đại diện của Apple và Mozilla đã bắt đầu thêm các mô-đun chuyển động CSS vào CSS3.

Hầu hết các trình duyệt không hỗ trợ CSS3 transition. Để các trình duyệt hỗ trợ CSS3 transition thì cần phải thêm vào css các tiền tố như:

- Apple safari 3.1 hoặc cao hơn thêm: - webkit

- Google Chrome: webkit

- Mozilla Firefox 3.7 alpha hoặc cao hơn: -moz-

Ví dụ: Sử dụng CSS3 transition để thay đổi độ rộng sau 3 giây.

```
div {  
  transition: width 3s;  
  
  -moz-transition: width 3s; /* Firefox 4 */  
  -webkit-transition: width 3s; /* Safari and Chrome */  
  -o-transition: width 3s; /* Opera */  
}
```

Danh sách các thuộc tính transition

Thuộc tính	Mô tả
transition	Là thuộc tính kế hợp. Mô tả tổng hợp, rút gọn
transition-property	Quy định giá trị chuyển động (theo chiều nào)
transition-duration	Định nghĩa thời gian, đơn vị giây (s)
transition-timing-function	Được sử dụng để mô tả tốc độ chuyển đổi sẽ được thực hiện như thế nào.
transition-delay	Được sử dụng để xác định sự bắt đầu của quá trình chuyển đổi. Giá trị mặc định là 0

4.8.3. CSS3 Animation

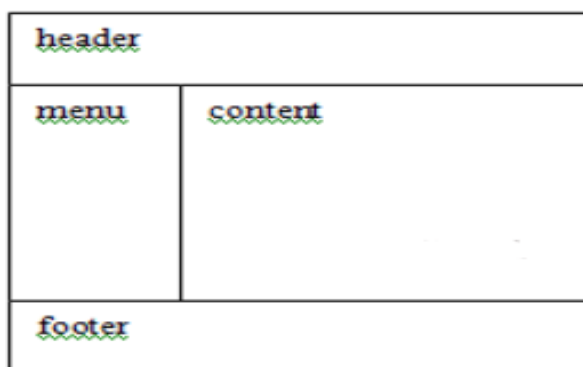
Hình ảnh động CSS3 cho phép hoạt hình của hầu hết các phần tử HTML mà không sử dụng JavaScript hoặc Flash!

4.9. KẾT HỢP THẺ `div` VÀ CSS ĐỂ TẠO BỐ CỤC TRANG WEB

Trước đây, để thiết kế giao diện (layout) cho website người ta phải sử dụng bảng (table). Nhưng khi sử dụng table, chúng ta viết các thẻ HTML khá dài khiến website load chậm, và có thể xảy ra các đoạn mã thừa không cần thiết, hơn nữa không thân thiện với các công cụ tìm kiếm như Google. Việc sử dụng DIV kết hợp với CSS để làm layout (bố cục) cho một trang web đã trở lên phổ biến và đạt hiệu quả. Vì vậy, xây dựng layout bằng thẻ DIV kết hợp với CSS là phương pháp phổ biến trong thiết kế web hiện nay thay cho phương pháp sử dụng thẻ TABLE.

a. Nguyên tắc dựng layout bằng cách dùng thẻ `<div>`

Dựng layout bằng cách phân vùng giao diện thành các khối chữ nhật DIV theo chiều từ trên xuống dưới, từ trái qua phải.



Hình 3-8. Nguyên tắc dựng layout

Với layout của hình 2.3 ta tạo các khối DIV như sau:

- ✓ DIV1: Chứa header
- ✓ DIV 2: Chứa menu và content. Trong DIV2 lại chia nhỏ thành 2 DIV (Div21 chứa menu và Div22 chứa content)
- ✓ DIV 3: Chứa footer

b. Cách điều khiển vị trí của các phần tử DIV

Để điều khiển vị trí của các phần tử DIV ta dùng thuộc tính Float và Clear. Đây là 2 thuộc tính rất cần thiết khi ta muốn cố định vị trí 1 khối bao quanh thẻ `<div>`.

❖ Thuộc tính Float: Thuộc tính float có 3 giá trị:

- Left: Cố định phần tử về bên trái.
- Right: Cố định phần tử về bên phải.
- None: Bình thường.

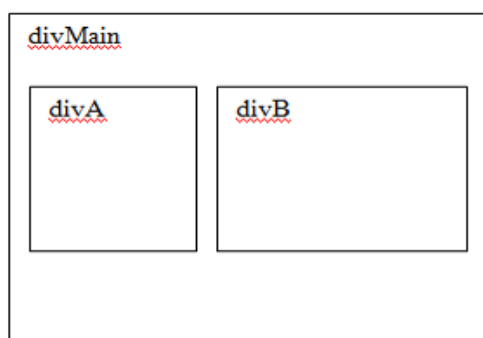
❖ Clear

Đi cùng với thuộc tính float, trong CSS còn có một thuộc tính là clear. Thuộc tính clear là một thuộc tính thường được gán vào các phần tử liên quan tới phần tử đã được float để quyết định hướng xử sự của phần tử này.

Thuộc tính Clear có 4 giá trị:

- left (tràn bên trái)
- right (tràn bên phải)
- both (không tràn)
- none.

Ví dụ: Giả thiết ta có 2 khối chữ nhật là divA và divB, được chứa trong khối chữ nhật divMain (minh họa ở hình 2.4).



```
<div id= "divMain">  
    <div id= "divA"> </div>  
    <div id= "divB"> </div>  
</div>
```

Hình 3-9. Điều khiển vị trí tương đối của divA, divB trong lòng divMain

✓ Muốn divA bám dính vào cạnh trái của divMain, còn divB bám dính cạnh phải của divMain ta sử dụng thuộc tính float cho phần tử divA và divB như sau:

```
#divA{  
    float: left;  
}  
#divB{  
    float: right;  
}
```

✓ Muốn chỉnh vị trí divA dịch vào sâu trong lòng divMain với một khoảng đo chính xác, sử dụng margin-left, margin-top.

Ví dụ: làm cho divA cách top của divMain 10px, cách left của divMain 10px, ta viết như sau:

```
#divA{  
    float: left;  
    margin-left:10px;  
    margin-top:10px;  
}
```

✓ Đẩy divB nằm xuống phía dưới divA thì dùng thuộc tính clear:

```
#divA{  
    float:left;
```

```
}  
#divB {  
clear:both;  
}
```

Thuộc tính **clear:both** làm cho DIV B không chấp nhận bất cứ khối DIV nào "chèn" hai đầu trái và phải của nó, do đó nó tụt xuống dưới divA để không bị divA "chặn" ở đầu trái của nó.

✓ Khi hiển thị divA và divB nằm cạnh nhau theo chiều ngang bên trong divMain như trong hình 2.4 thì phải đảm bảo:

$\text{Độ rộng (DivA)} + \text{Độ rộng (DivB)} \leq \text{Độ rộng (DivMain)}$

✓ Khi tính độ rộng của một DIV cần tính gộp cả vào đó độ rộng của margin-left, margin-right, border-left, border-right

Ví dụ: định nghĩa CSS của divA

```
#divA {  
margin-left:10px;  
width:800px;  
border-left:1px ;  
border-right:1px ;}
```

Độ rộng của divA khi đó là: $800 + 10 + 1 + 1 = 812(\text{px})$. Cần lưu ý điều này để tránh bị vỡ DIV khi ghép các DIV theo chiều ngang.

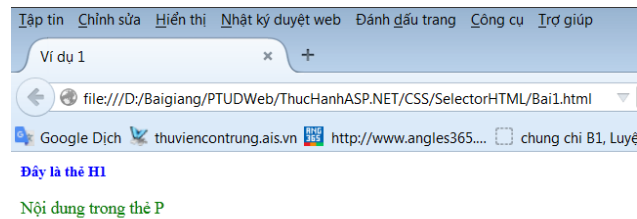
4.8 MỘT SỐ VÍ DỤ

Ví dụ 1: Đoạn mã dưới đây định nghĩa thuộc tính màu 'color' và font-size cho phần tử **H1** và **H2**:

```
<HTML>  
<HEAD>  
<TITLE>Ví dụ 1</TITLE>  
<meta charset="utf-8">  
<STYLE TYPE="text/css">  
    H1 { font-size: 12px; color: Blue; }  
    P { font-size: 14px; color: green; }  
</STYLE>  
</HEAD>  
<Body>  
<H1> Đây là thẻ H1 </H1>  
<P> Nội dung trong thẻ P </P>  
</Body>
```

<HTML>

Kết quả:



Hình 3-10: Sử dụng thuộc tính màu color và font-size cho H1, H2

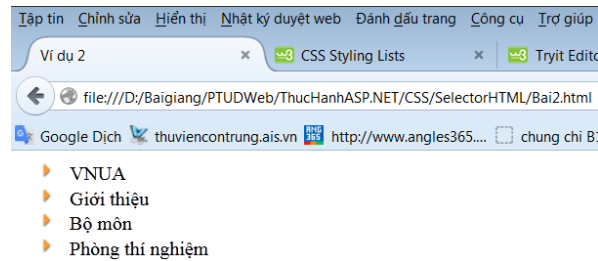
Ví dụ 2: Dùng style sheet để định dạng danh sách

```
<HTML>
<HEAD>
  <TITLE>Ví dụ 2</TITLE>
  <meta charset="utf-8">
  <STYLE TYPE="text/css">
    UL#menungang{list-style-image:url('images/1arrow_forward_1.gif'); }
    UL#menudoc{list-style-type: square;}
    UL#menudoc li {display:inline;}
  </STYLE>
</HEAD>
<BODY>
  <UL id="menungang">
    <LI>VNUA</li>
    <LI>Giới thiệu</li>
    <LI>Bộ môn</li>
    <LI>Phòng thí nghiệm</li>
  </UL>
  <H1>Menu đọc</H1>
  <UL id="menudoc">
    <LI>Lời giới thiệu</li>
    <LI>Sinh viên</li>
    <LI>Giáo trình</li>
    <LI>Liên hệ</li>
  </UL>
```



```
</UL>
</BODY>
</HTML>
```

Kết quả:



Menu dọc

Lời giới thiệu Sinh viên Giáo trình Liên hệ

Hình 4.7: Sử dụng CSS để định dạng danh sách

Ví dụ 3: Dùng style sheet để định dạng danh sách tạo menu ngang có màu Blue, khi rê chuột vào từng mục của menu chuyển sang màu cam

```
<HTML>
<HEAD>
  <TITLE>Ví dụ 2</TITLE>
  <meta charset="utf-8">
  <STYLE TYPE="text/css">
    UL#menudoc{list-style-image:url('images/1arrow_forward_1.gif');
  }

  UL#menungang li {display:inline;}
  ul#menungang {padding: 0;}
  ul#menungang li a{
    background-color: Blue;
    color: white;
    padding: 10px 20px;
    text-decoration: none;
    border-radius: 4px 4px 0 0;
```

```
    }

    ul#menungang li a:hover {
        background-color: orange;
    }
</STYLE>
</HEAD>
<BODY>
    <UL id="menudoc">
        <LI>VNUA</li>
        <LI>Giới thiệu</li>
        <LI>Bộ môn</li>
        <LI>Phòng thí nghiệm</li>
    </UL>
    <H3>Menu dọc có màu BLUE, khi rê chuột vào từng mục chuyển màu
    ORANGE</H3>
    <UL id="menungang">
        <LI><a href="Gioithieu.html"> Lờì giới thiệu</a></li>
        <LI><a href="sinhvien.aspx">Sinh viên</a></li>
        <LI><a href="giaotrich.html">Giáo trình</a></li>
        <LI><a href="lienhe.html">Liên hệ</a></li>
    </UL>
</BODY>
</HTML>
```

Kết quả:



Hình 4.8: Sử dụng CSS để định dạng danh sách cho menu ngang

Ví dụ 4: Dùng thẻ div để tạo layout có cấu trúc như hình dưới:



Code: Để tạo được layout như trên ta thiết lập 2 file (1file .html và 1 file .css)

- Nội dung file .html

```
<html>
  <head>
    <meta charset="utf-8" />
    <title>Tiêu đề của trang</title>
    <link rel="stylesheet" href="css\style_div.css" />
  </head>
  <body>
    <div class="bao">
```

```
<div class="banner">
</div>
<div class="cottrai">
    <br>
    <li>Trang chủ</li>
    <li>Tổ chức</li>
    <li>Đào tạo</li>
    <li>Nghiên cứu</li>
    <li>Bộ môn </li>
    <li>Sinh viên</li>
    <li>Đoàn thể</li>
    <li>Cơ sở vật chất</li>
</div>

<div class="noidung">
    <p align=center>Đây là phần nội dung chính</p>
</div>
<div class="footer">
    <center>Khoa CNTT- Đại học Nông nghiệp Hà
Nội<br>
    Địa chỉ: Trâu Quỳ, Gia Lâm, Hà Nội
    <br>Tel: 04.8766106, Fax: 04.276554
    </center>
</div>
</div>
</body>
</html>
```

- Nội dung file .css:

```
.bao {
    width: 100%;
}
.banner {
    width: 100%;
```

```
background:#A9A9A9;
float: right;
display: block;
background-image:url('images/banner.png');
height: 170px;
}

.cottrai {
width: 20%;
float: left;
display: block;
background:#DCDCDC ;
height: 800px;
}

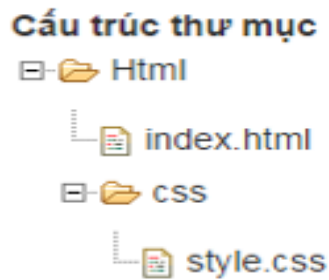
.noidung {
width: 80%;
float: left;
display: block;
background-image:url('images/bg.jpg');
height: 800px;
}

.footer {
width: 100%;
float: left;
background:#6495ED;
height:100px;
}
```

CÂU HỎI VÀ BÀI TẬP CHƯƠNG 4

Nếu như chương 1 chúng ta đã sử dụng các thẻ của HTML để xây dựng cấu trúc trang web thì phần này thay vì sử dụng các thẻ HTML ta sẽ sử dụng CSS để định dạng font chữ, màu chữ,... cho các trang web đã xây dựng.

Cách thức tổ chức dữ liệu:

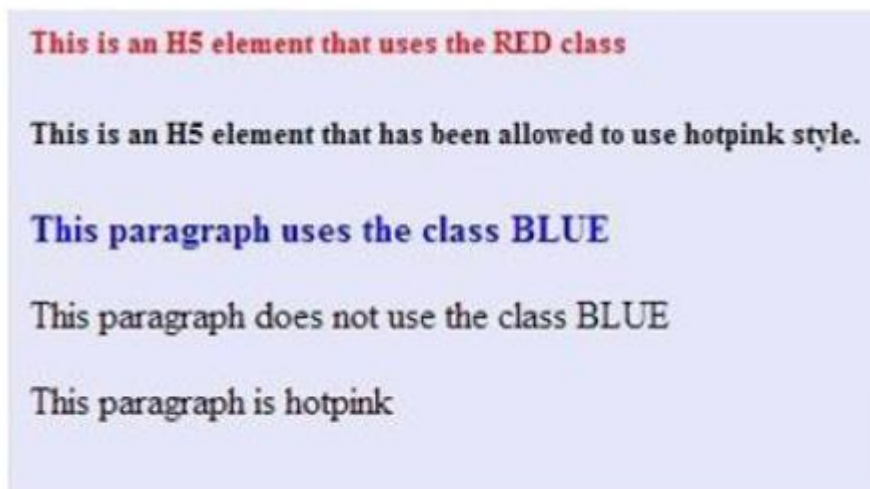


Bài 1: Tạo một trang web động dùng style sheet hiển thị như hình bên dưới:



Bài 2: Sử dụng CSS và mã HTML tạo một trang với các yêu cầu như sau:

- Có giao diện như hình vẽ
- Áp dụng CSS cho các phần tử trong bài tập trên bằng ít nhất 3 cách viết.

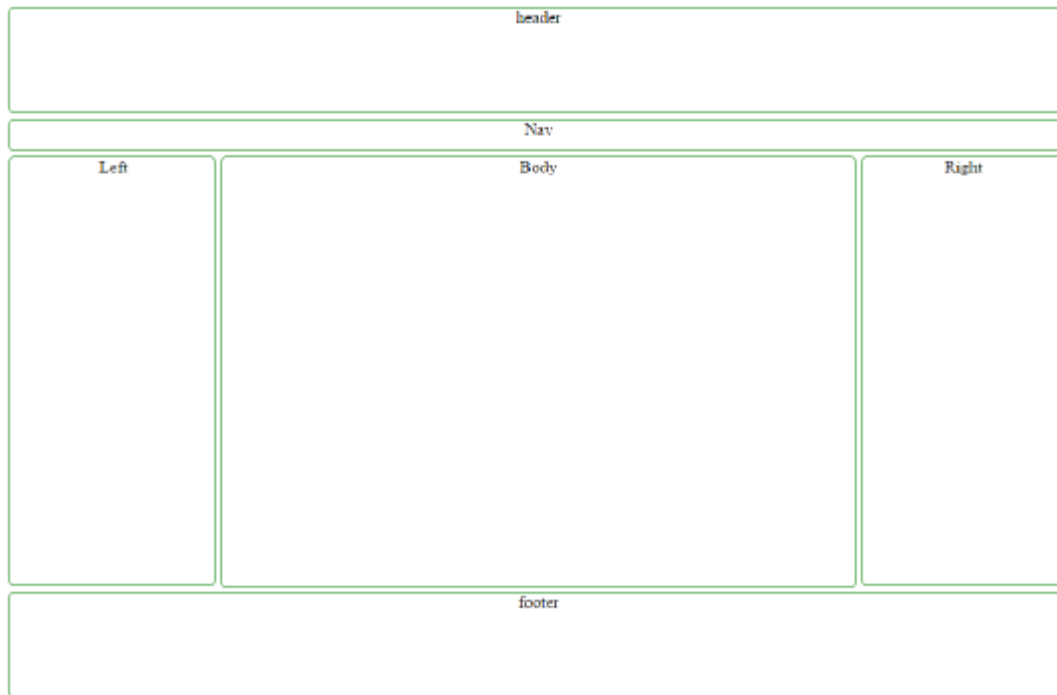


Bài tập 3: Viết chương trình thực hiện:

- Tạo trang login (Sử dụng HTML5 và CSS, CSS3) – dangnhap.html (Ví dụ như google.com)
- Tạo trang đăng ký (dangky.html) – sử dụng công cụ HTML5 và CSS3 (Ví dụ như google.com)

Bài tập 4: Viết chương trình thực hiện:

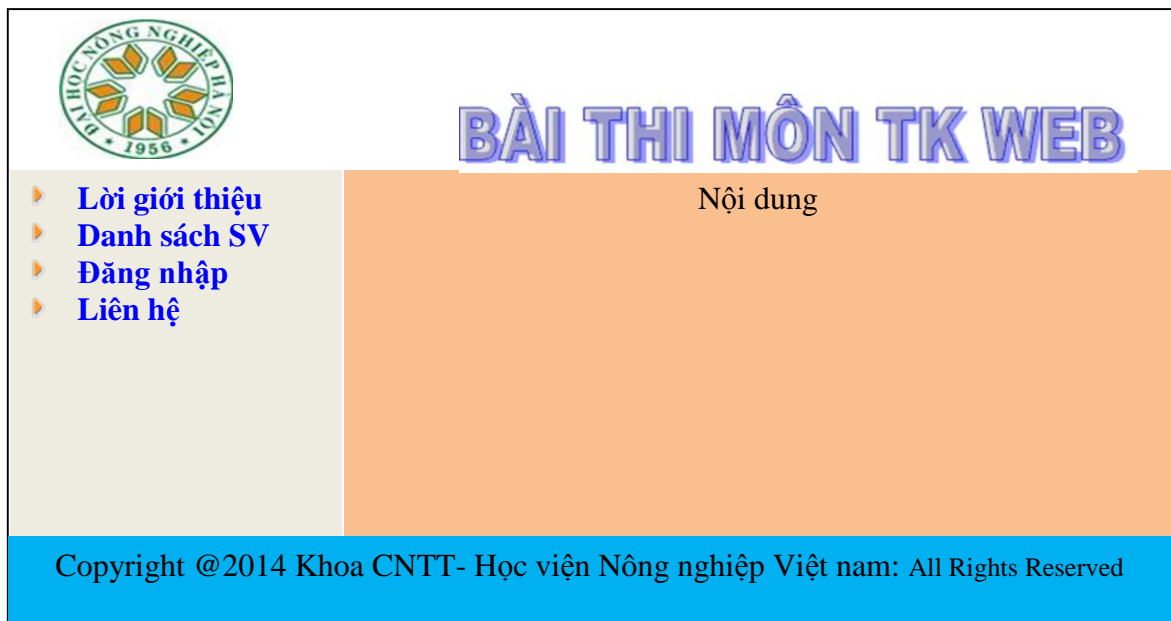
- Tạo giao diện web (trang mẫu) như hình dưới



- Yêu cầu: Sử dụng HTML5 và CSS3

Bài 5: Dùng HTML và CSS để thiết kế các trang web có giao diện như mẫu 1:

Mẫu 1:



Bài 6: Dùng HTML và CSS để thiết kế các trang web có giao diện như mẫu 2:

Mẫu 2:



Bài 7: Dùng HTML và CSS để thiết kế các trang web có giao diện như mẫu 3:

Mẫu 3:



Bài 8: Dùng HTML và CSS để thiết kế các trang web có giao diện như mẫu 4:

Mẫu 4:



TÀI LIỆU THAM KHẢO CHƯƠNG 4

1. Jon Duckett (2014). *HTML & CSS Design and Build Websites*. Wiley.
2. <http://www.w3schools.com/css/default.asp>