KHOA CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG
BỘ MÔN MẠNG MÁY TÍNH VÀ TRUYỀN THÔNG

LẬP TRÌNH WEB

Chương 4: CSS

ThS. NGUYỄN CAO HỒNG NGỌC

NỘI DUNG

- Giới thiệu CSS
- Cú pháp CSS
- Vị trí đặt CSS
- Nguyên tắc áp dụng
- Thuộc tính CSS
- Các thiết kế nâng cao

GIỚI THIỆU CSS

- CSS viết tắt từ Cascading Style Sheets
- HTML được thiết kế để mô tả cấu trúc, nội dung của trang web
- CSS được thiết kế để định dạng cách hiển thị nội dung của trang web
- Các kiểu (styles) xác định cách thức trình bày các phần tử html
- Các kiểu được bổ sung vào HTML 4.0 để giải quyết vấn đề về hiển thị
- Việc dùng các bảng kiểu ngoài (External Style Sheets) làm giảm khối lượng công việc đáng kể
- Các bảng kiểu ngoài được lưu vào các tập tin có phần mở rộng là *.css*

Tại sao là CSS?

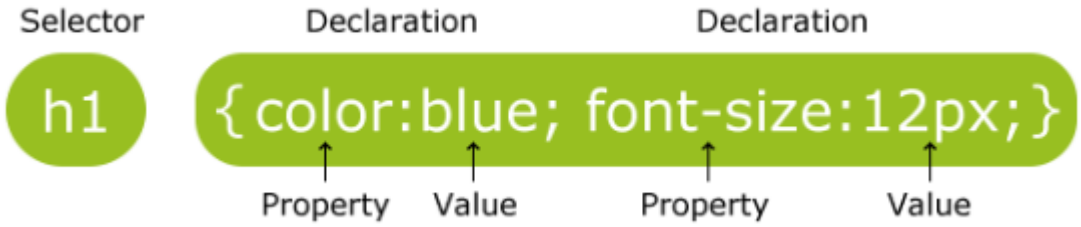
- HTML không chứa các thẻ định dạng tài liệu
- HTML chỉ định nghĩa nội dung của tài liệu dưới các dạng như:

```
<h1>This is a heading</h1>
<p>This is a paragraph</p>
```

- Khi thẻ <font>, và thuộc tính color được thêm tới đặc tả HTML 3.2, nó phát sinh ra vấn đề làm đau đầu các nhà phát triển web: Việc phát triển các website lớn, với thẻ font và thuộc tính color thêm vào mỗi trang, trở nên tốn kém thời gian và chi phí nhiều hơn
- Để giải quyết vấn đề này, World Wide Web Consortium (W3C) tạo ra CSS
- Trong HTML 4.0, tất cả các định dạng được được gỡ khỏi tài liệu HTML, và được lưu vào một tập tin CSS riêng.
- Tất cả các trình duyệt ngày nay đều hỗ trợ CSS.

CÚ PHÁP CSS

- Cú pháp cơ bản: Một luật css (css rule) có 2 phần: một Selector, và một hay nhiều khai báo



- Trong đó:
  - Selector (bộ chọn): xác định các thành phần HTML sẽ được áp dụng kiểu tương ứng
  - Property (thuộc tính): các thuộc tính định dạng
  - Value: giá trị các thuộc tính tương ứng
- Mỗi khai báo chứa một cặp thuộc tính & giá trị
- Thuộc tính & giá trị được ngăn cách nhau bởi dấu ":", các khai báo được ngăn cách nhau bởi dấu ";".
- Mỗi khai báo nên đặt trên từng dòng

Cú pháp cơ bản (tt)

- Nhóm các bộ chọn: khi một kiểu được áp dụng cho nhiều bộ chọn ta có thể kết hợp chúng lại với nhau. Cú pháp:

```
selector1, selector2,… {property:value}
```

Ví dụ:

```
h1{color:red;} h2{color:red;} h3{color:red;}
h1,h2,h3{color:red;}
```

- Chú thích trong CSS bắt đầu với "/*" và kết thúc với "*/"

Ví dụ:

```
/* Author: Jason Cranford Teague
   Date:   08/20/2010                    */
p{
    text-align:center
    color:black;  /* Font color is black */
    font-family:arial;
}
```

Các loại bộ chọn

- Có 4 loại bộ chọn cơ bản:
  - HTML Selector
  - Class Selector
  - ID Selector
  - Attribute Selector
- HTML Selector:
  - Bộ chọn chính là tên các thẻ trong HTML
  - Kiểu sẽ áp dụng cho tất cả các thẻ tương ứng trên toàn bộ trang

    Ví dụ: Với khai báo CSS sau toàn bộ nội dung của các thành phần h2 trong trang web sẽ có font chữ màu đỏ

Class Selector

- Với bộ chọn class có thể định nghĩa các kiểu cho một hay một nhóm các thành phần html
- Các kiểu với bộ chọn class sẽ chỉ được áp dụng cho các thành phần html có thuộc tính class với giá trị là tên bộ chọn class tương ứng

Ví dụ:

Để áp dụng các kiểu của *bộ chọn class* trên cho thành phần *h2* trong trang html ta thực hiện như sau:

*<h2 **class**="hilight">Chapter I…</h2>*

Class Selector (tt)

- **Dependent Class:** cho phép xác định các kiểu của một class chỉ có thể áp dụng trên một loại thẻ HTML => khả năng tạo một kiểu chung cho một class, và định nghĩa các kiểu riêng cho các thẻ HTML xác định trong class đó

  Ví dụ:

*<h2 **class**="hilight">Chapter I…</h2>* => Green Background

*<p **class**="hilight">I should…</p>* => Yellow Background

- **Trộn và ghép các class:** có thể thêm nhiều class vào một thẻ HTML để trộn và ghép các kiểu cần sử dụng, bằng cách đưa tất cả các class cần vào thuộc tính class và phân cách nhau với ký tự khoảng trắng.

  Ví dụ: *<p **class**="hilight smallprint">I should…</p>*

ID Selector

- Bộ chọn ID dùng để chỉ định kiểu cho duy nhất một thành phần HTML
- Các kiểu của bộ chọn ID chỉ được áp dụng cho duy nhất các thành phần HTML có thuộc tính ID với giá trị là tên bộ chọn ID tương ứng

Ví dụ:

Để áp dụng các kiểu của *bộ chọn ID* trên cho thành phần *h2* trong trang html ta thực hiện như sau:

*<h2 **id**="title">Chapter I...</h2>*

Qui tắc đặt tên cho bộ chọn class & id

- Bắt đầu bằng một ký tự A-Z hay a-z
- Theo sau bởi các ký tự (A-Z, a-z) , số, hay các ký tự "-", "_"
- Tên phân biệt HOA hay thường

  Ví dụ: Các tên hợp lệ: *greenText*, *Font34*, *some_style*, *_newStyle*, *-italicStyle*

  Các tên không hợp lệ: *@RedBorder*, *4_xyz*
- ***Lưu ý:*** *nên đặt các tên có ý nghĩa*

Attribute Selector

- Xác định kiểu cho các thành phần HTML có một thuộc tính xác định
- **Lưu ý:** IE7 và IE8 chỉ hỗ trợ attribute selector khi có khai báo !DOCTYPE. IE6 và các phiên bản IE thấp hơn không hỗ trợ attribute selector

Selector	Example	Example description
[attribute]	[target]	Selects all elements with a target attribute
[attribute=value]	[target=_blank]	Selects all elements with target="_blank"
[attribute~=value]	[title~=flower]	Selects all elements with a title attribute containing the word "flower"
[attribute|=language]	[lang|=en]	Selects all elements with a lang attribute value starting with "en"

- *Kết hợp giữa HTML Selector & Attribute Selector:*
  - Xác định kiểu cho một loại thành phần HTML có một thuộc tính xác định

    Ví dụ: input[type="text"] {background-color:blue}

    ⇒ tất cả phần tử nhập (input) mà có thuộc tính kiểu với giá trị là "text" sẽ có màu nền là màu xanh

Kiểu trong ngữ cảnh

- **Descendent Selector Context:** Các kiểu có thể được áp dụng cho một thành phần HTML tùy thuộc vào các thẻ HTML, class hay id của các thành phần cha của chúng

Ví dụ 1:

```
h1 cite { color: red; }
```

Space

⇒ nếu một thành phần cite nằm trong thành phần h1, thì văn bản của nó có màu đỏ

Ví dụ 2:

```
#article p.intro cite { color: red; }
```

Kiểu trong ngữ cảnh (tt)

- **Styles for Children:** Xác định kiểu cho một thành phần html là *con trực tiếp* của một thành phần html khác

Ví dụ:

```
p>em{ color: red; }
```

Chevron

⇒ `<p><em> the white kitten</em></p>`

- **Styles for Siblings:** xác định kiểu cho một thành phần html cùng cấp kế cận tiếp theo với thành phần html hiện hành

Ví dụ:

```
em+cite{ color: red; }
```

Plus Sign

⇒ `<em> Quotes</em> from`
`<cite>Through the Looking-Glass</cite>`

Kiểu cho các trường hợp đặc biệt

- **Link pseudo-classes:** xác định kiểu cho các hypertext link
  - Các trạng thái có thể có của một liên kết:
    - **Link**: trạng thái khi liên kết chưa được kích hoạt
    - **Hover**: trạng thái khi người dùng di chuyển chuột qua
    - **Active**: trạng thái khi người dùng click chuột vào
    - **Visited**: trạng thái khi liên kết đã được kích hoạt.
  - Cú pháp: **<selector>:<link_state>{property:value;}**

Ví dụ:

```
a:link{color:red;}
a:visited{color:darkred;}
a:hover{color:hotpink; }
a:active{color:fuchsia;}
```

⟹ *<a href="index.html">We l c o m e ...</p>*

Kiểu cho các trường hợp đặc biệt (tt)

- **Dynamic pseudo-classes:** xác định kiểu cho bất kỳ thành phần nào phụ thuộc vào cách người dùng tương tác với nó
  - Các trạng thái: hover, active, focus (trạng thái khi người dùng chọn một thành phần trên màn hình)

Ví dụ:

```
input.formField{color:gray;}
input.formField:hover{color:green;}
input.formField:active{color:red;}
input.formField:focus{color:black;}
```

⇒ <input class="formField" type="text" />

Kiểu cho các trường hợp đặc biệt (tt)

- **Pseudo-elements:** xác định kiểu cho ký tự đầu tiên (*first-letter*) hay dòng đầu tiên (*first-line*) trong một khối văn bản
  - Cú pháp: **<selector>:<pseudo-elements>{property:value;}**

Ví dụ:

```
p:first-letter{color:red;}
p:first-line{color:blue;}
```

⇒ * You can use the :first-line pseudo-element to add a special effect to the first line of a text. *

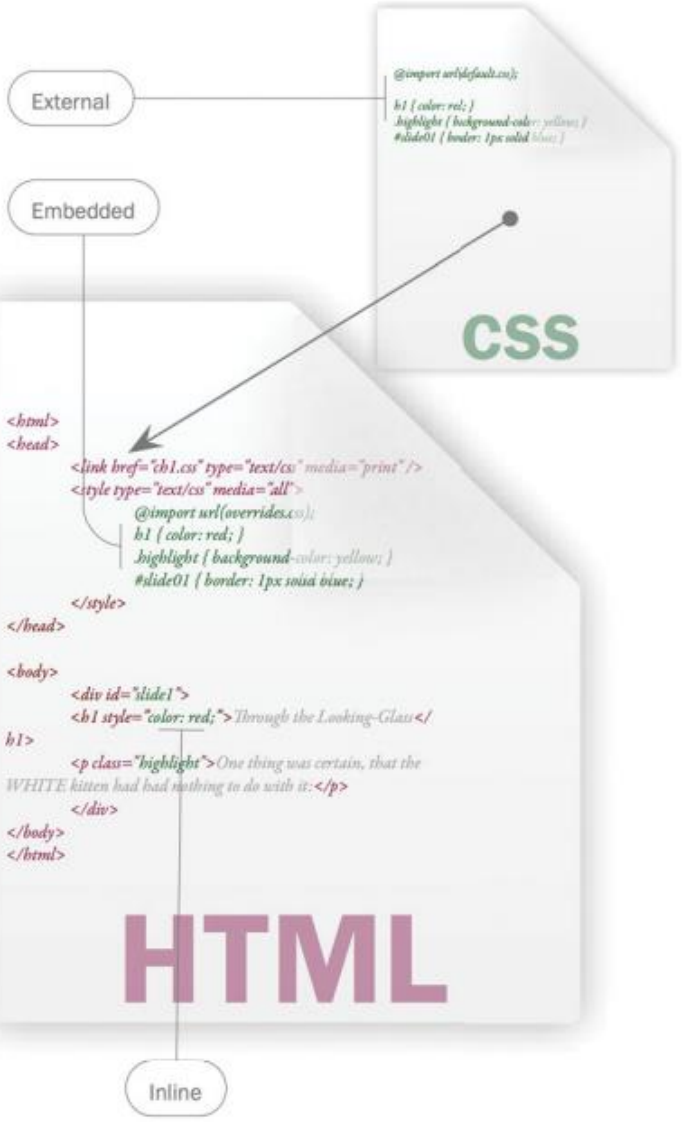
All CSS Pseudo Classes/Elements

Selector	Example	Example description
:link	a:link	Selects all unvisited links
:visited	a:visited	Selects all visited links
:active	a:active	Selects the active link
:hover	a:hover	Selects links on mouse over
:focus	input:focus	Selects the input element which has focus
:first-letter	p:first-letter	Selects the first letter of every <p> element
:first-line	p:first-line	Selects the first line of every <p> element
:first-child	p:first-child	Selects every <p> elements that is the first child of its parent
:before	p:before	Insert content before every <p> element
:after	p:after	Insert content after every <p> element
:lang(*language*)	p:lang(it)	Selects every <p> element with a lang attribute value starting with "it"

- Tất cả các loại bộ chọn: http://w3schools.com/cssref/css_selectors.asp

VỊ TRÍ ĐẶT CSS

- Inline style: các luật css được đặt trực tiếp vào một thẻ HTML sử dụng thuộc tính *style*
- Internal style sheet (embedded): các luật css được đặt trong thẻ <style> để định dạng cho trang hiện hành
- External style sheet: Các luật css được đặt trong các file riêng biệt (.*css*) có thể được truy cặp bởi bất kỳ trang web nào bằng cách dùng thẻ <link> hay lệnh *@import*



Inline style

- Các khai báo được để trực tiếp vào một thẻ HTML thông qua thuộc tính *style*
- Không cần bộ chọn, các kiểu chỉ được áp dụng cho thành phần HTML chứa thuộc tính style tương ứng
- Mặc dù hữu ích để nhanh chóng thêm vào các kiểu tại nơi cần thiết, nhưng inline style có 2 bất lợi:
  - Không thể tạo các kiểu định dạng chung cho toàn trang
  - Các kiểu trong inline style không thể bị đè lên

Ví dụ:

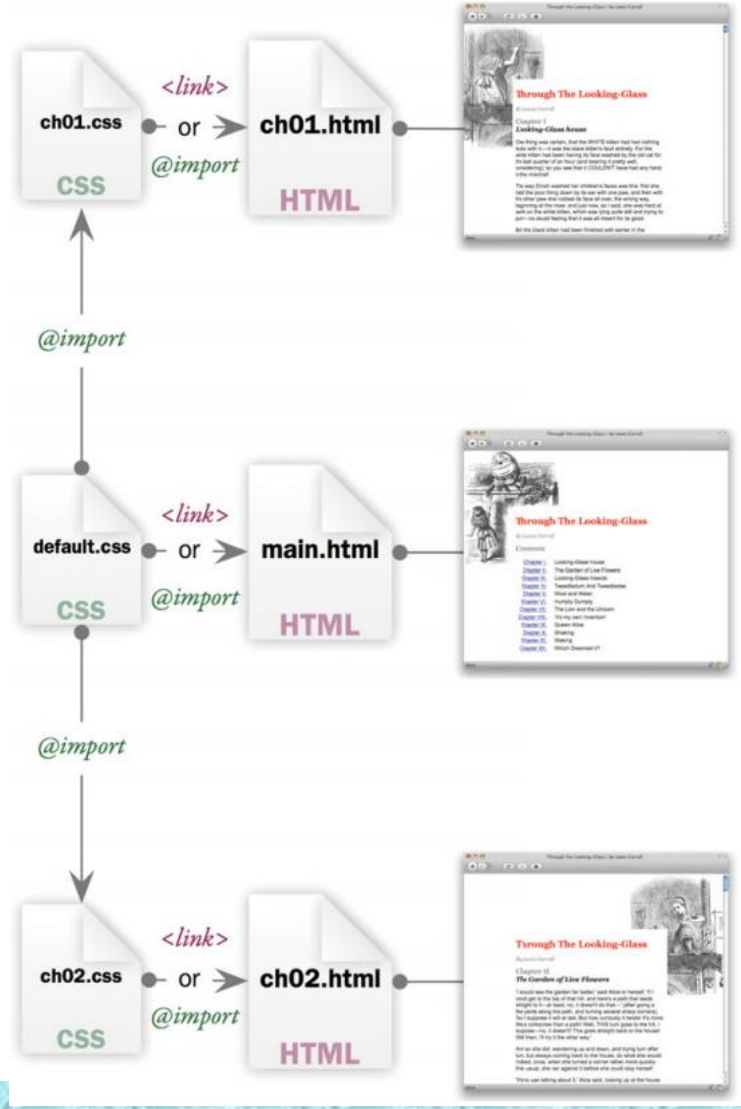
```
<h1 style="color: red; font-family: Georgia; text-align: center;">
Through the Looking-Glass</h1>
```

Internal style sheet

- Các kiểu sẽ có phạm vi ảnh hưởng trên toàn bộ trang web chứa nó
- Các luật css nằm trong cặp thẻ <style> … </style>
-
```
<style type="text/css" media="all">
    h1{color:red;}
    .hilight{background-color:yellow;}
    #slide01{border:1px solid blue;}
</style>
```

External style sheet

- Có thể định dạng cho toàn bộ web site bằng cách dùng các bảng kiểu ngoài
- File css là các file chỉ chứa mã CSS (không có thẻ <style>)
- Có thể dùng bất cứ trình soạn thỏa thuần văn bản nào để soạn thảo các file css
- Sử dụng <link> hoặc lệnh @import trong HTML file để trỏ đến các file css cần dùng cho một trang web



External style sheet (tt)

- <link>: nên đặt trong thẻ <head> và trước mã lệnh JavaScript

<link href="URL" type="text/css" media="all" />

Ví dụ: *<link href="default.css" type="text/css" media="all" />*

- @import: đặt trong thẻ style và trước bất kỳ mã CSS nào, nếu dùng trong CSS file thì được đặt ở đầu file

@import url(URL)

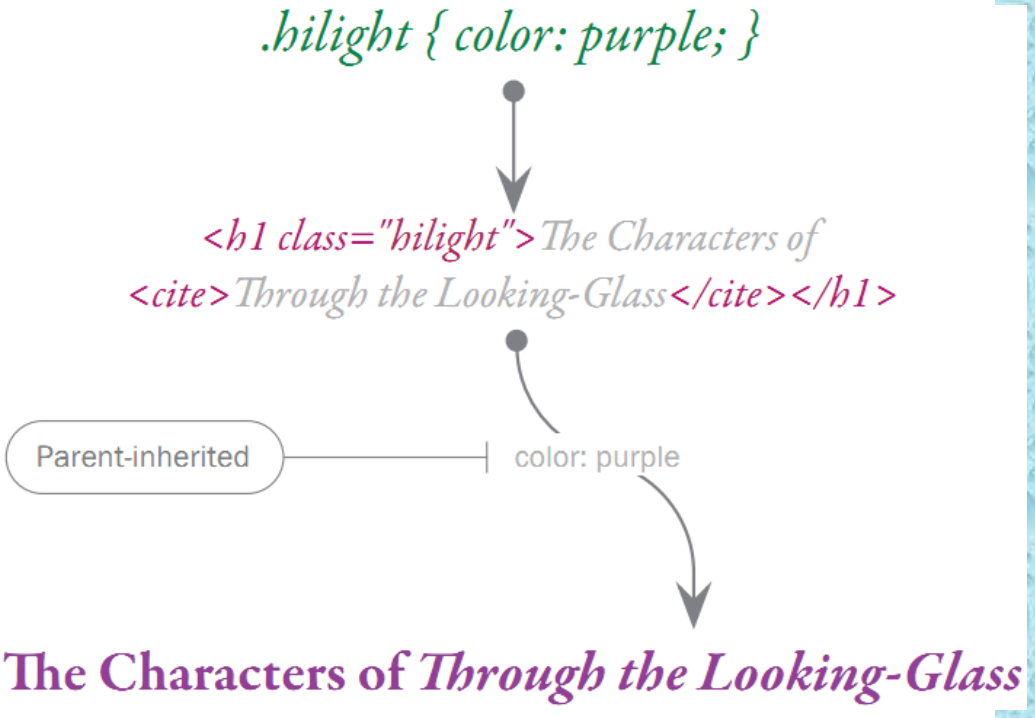
Ví dụ: *@import url(default.css);*

- Có thể dùng nhiều thẻ <link> hay lệnh @import để trỏ đến các file css cần dùng cho một trang html
- Các kiểu được xác định trong bảng kiểu ngoài sẽ ảnh hưởng đến bất cứ trang nào *link* hay *import* nó.

NGUYÊN TẮC ÁP DỤNG CSS

- Việc áp dụng một kiểu định dạng cho một thành phần HTML tuân theo các luật sau: Inline styles, Media, Importance, Specificity, Order, Parent-inherited, Browser-default

- **Inheritance** (sự thừa kế): hầu như tất cả các thẻ HTML điều bị ảnh hưởng bởi các kiểu định dạng không trực tiếp. Các kiểu này có thể là kiểu mặc định của trình duyệt, hay bị ảnh hưởng bởi kiểu của các thẻ cha (parent tags). Các kiểu được thừa kế thì dễ dàng được ghi đè bằng việc dùng các bảng kiểu css.

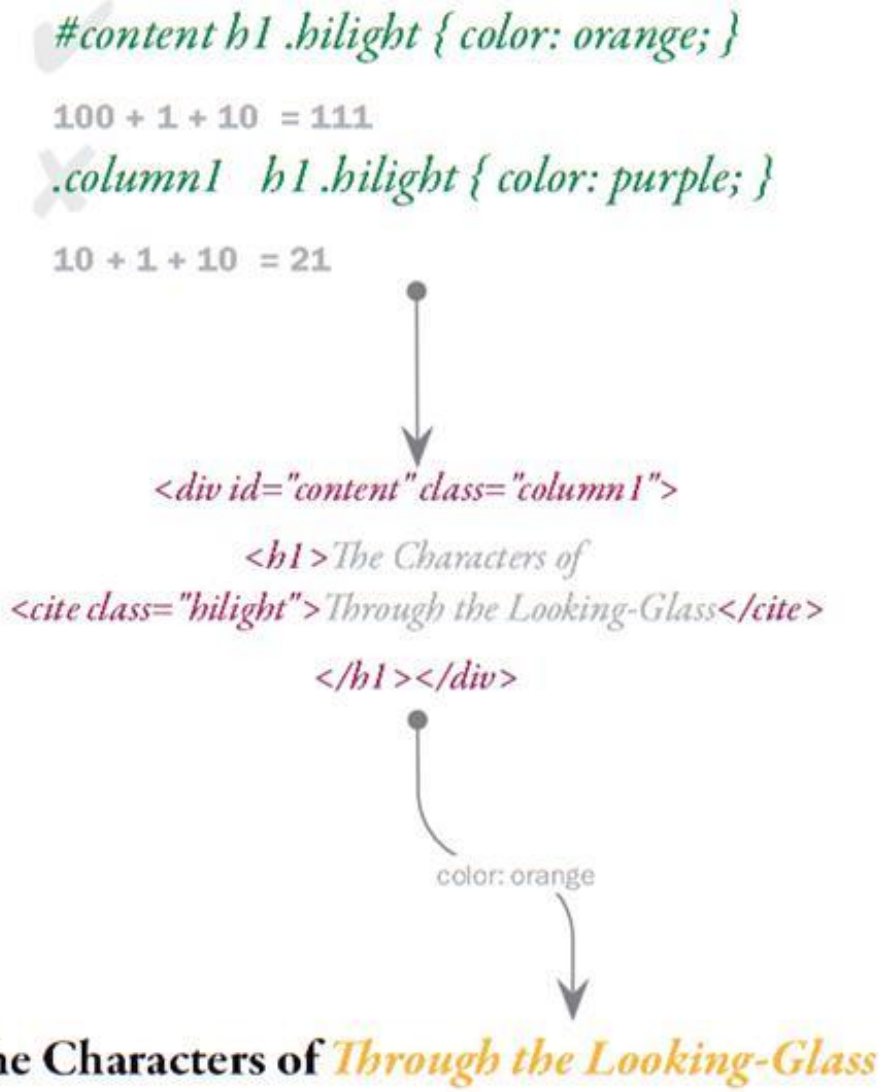


Order (thứ tự)

- Có thể dễ dàng viết đè lên một kiểu của bất kỳ bộ chọn nào bằng cách khai báo lại bộ chọn với các thuộc tính giống nhau nhưng khác giá trị => các kiểu khai báo sau cùng sẽ được áp dụng vào trang

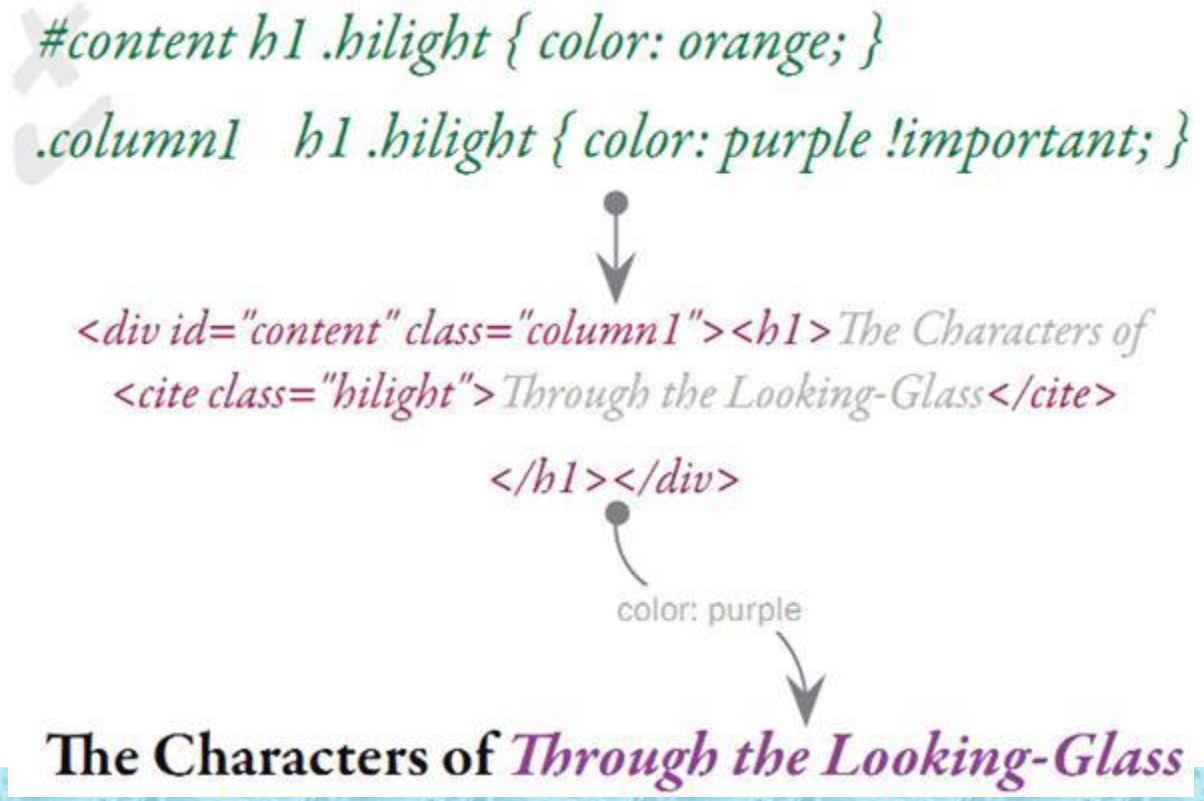
Specificity

- **Specificity (đặc trưng):** xác định thứ tự ảnh hưởng của các kiểu phụ thuộc vào bộ chọn ngữ cảnh => mức độ đặc trưng của bộ chọn càng cao độ ưu tiên các kiểu của nó được áp dụng cho một phần tử càng cao (bất kể thứ tự của bộ chọn)
  - Mức độ đặc trưng của bộ chọn ngữ cảnh phụ thuộc vào các kiểu bộ chọn được sử dụng trong bộ chọn ngữ cảnh, mỗi bộ chọn có một trọng số ưu tiên riêng:
    - Bộ chọn HTML: 1
    - Bộ chọn Class & Attribute: 10
    - Bộ chọn ID: 100
    - Inline style: ∞



Importance

- Có thể dùng cú pháp *!important* để xác định một kiểu đặc biệt nào đó sẽ ưu tiên được sử dụng khi có sự cạnh tranh thứ tự áp dụng các kiểu css lên một thành phần html => mức độ ưu tiên cao hơn các luật inheritance, order và specificity.



Media

- Media hay phương tiện dùng để trình bày nội dung một trang web
- Các trang web có thể được xuất ra các thiết bị khác nhau, có 4 giá trị cho thuộc tính *media*:
  - *Screen*: màn hình máy tính cá nhân (laptop, desktop computer) gồm các loại: CRT, LCD, hay Plasma
  - *Print*: máy in
  - *Handheld*: các thiết bị cầm tay
  - *All*: các kiểu sẽ được sử dụng mà không xem xét kiểu thiết bị xuất
- Nếu kiểu thiết bị xuất không tương ứng với kiểu thiết bị đã chỉ định trong thuộc tính *media* thì các kiểu này sẽ bị bỏ qua

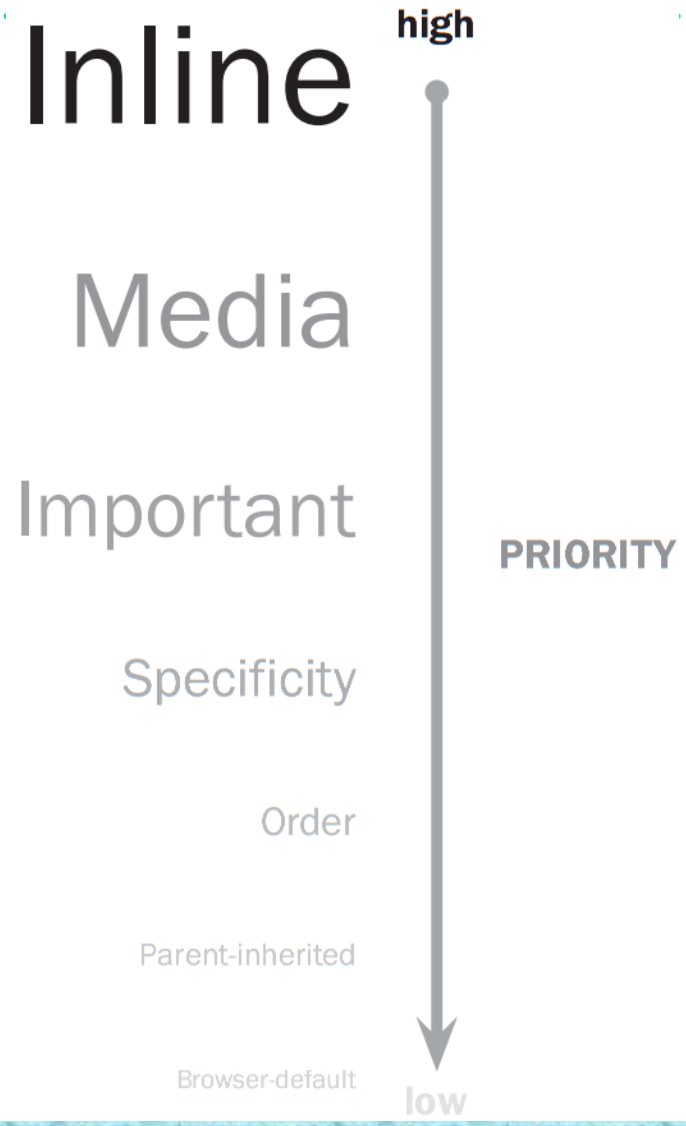
Screen

Print

Handheld

Độ ưu tiên áp dụng các kiểu CSS (Cascade)

- **Cascade (phân tầng):** việc áp dụng các kiểu css theo nguyên tắc ưu tiên, theo thứ tự các luật sau: Inline styles, Media, Importance, Specificity, Order, Parent-inherited, Browser-default



CSS PROPERTIES

1. Values
2. Font
3. Text
4. Background
5. Box model
    a. **Display & Visibility**
    b. **Float**
    c. **Width and Height**
    d. **Border**
    e. **Padding**
    f. **Margin**
    g. **Outline**
6. Position
7. Table
8. Lists
9. Cursor

1. Values

- Giá trị của một thuộc tính CSS gồm có 2 dạng: *keyword*, *variable*
- Keyword: các giá trị được gán cho một thuộc tính css có thể là một *keyword*. Một số keyword đặc biệt:
  - *Auto:* trình duyệt tự động tính giá trị
  - *Inherit:* thuộc tính sẽ dùng cùng giá trị với thành phần cha
  - *None:* gỡ bỏ các giá trị đã được gán trước đó
  - *Normal:* sử dụng giá trị mặt định của một thuộc tính
  - *Transparent:* xác định độ mờ đục (opacity) của giá trị màu là 0%, cho phép một thành phần có thể được nhìn xuyên qua.
- Variable: cần xác định một giá trị chính xác dựa vào kiểu của biến.
  - *Qui ước*: giá trị của một thuộc tính ở dạng biến được biểu diễn như sau: <type_of_variable>, **Ví dụ:** <color>

2. Fonts

- Các thuộc tính CSS font định nghĩa: font family, boldness, size, và kiểu của văn bản
- Sự khác nhau của font Sans-serif và Serif:

- Trên màn hình máy tính, các font Sans-serif được xem là dễ đọc hơn font Serif
- *font*: là một shorthand property cho phép đặt tất cả các thuộc tính font bằng cách liệt kê các giá trị cùng lúc như sau:
  - VALUES: inherit | <font-style> <font-variant> <font-weight> <font-size>/<line-height> <font-family>

2. Fonts (tt)

- Font Families:
  - *font-family*: xác định họ font thông qua các tên (font family name) như: "Serif" hay "Times New Roman"
  - Có 2 loại font family name:
    - *Generic family*: dùng để chỉ một nhóm các họ font gần giống nhau như: "Serif" hay "Monospace"
    - *Font family*: xác định một họ font đặt biệt như: "Times New Roman" hay "Arial"

Generic family	Font family	Description
Serif	Times New Roman Georgia	Serif fonts have small lines at the ends on some characters
Sans-serif	Arial Verdana	"Sans" means without - these fonts do not have the lines at the ends of characters
Monospace	Courier New Lucida Console	All monospace characters have the same width

2. Fonts (tt)

- Font Families (tt):
  - Thuộc tính *font-family* có thể quản lý nhiều tên font như hệ thống "fallback ".
  - Nếu browser không hỗ trợ font thứ nhất, nó thử font kế
  - Bắt đầu với font bạn muốn, và kết thúc với font họ chung, để browsers chọn một font tương tự trong họ chung nếu không có font nào khác sẵn có
  - Nếu tên của họ font nhiều hơn một từ, nó phải đặt trong dấu nháy kép: " "
  - Nhiều hơn một họ font được xác định trong danh sách được ngăn cách bởi dấu phẩy ","

    Ví dụ: *font-family: garamond, georgia, serif;*

    Garamond, Georgia, Serif

2. Fonts (tt)

- Font Style:
  - *font-style*: dùng để định dạng văn bản in nghiêng (italic) hay xiên (oblique)
  - VALUES : normal | italic | oblique | inherit
  - italic và oblique rất giống nhau, nhưng oblique ít được hỗ trợ hơn

Ví dụ:

font-style: italic;

Italic *oblique*

- Font Size:
  - *font-size*: xác định cỡ chữ của văn bản
  - VALUE: <length> | <percentage-parents-font-size> | smaller | larger | xx-small | x-small | small | medium | large | x-large | xx-large | inherit

Ví dụ:

font-size: 12px;

6px 12px 18px 24px

2. Fonts (tt)

- *font-weight*: xác định độ đậm của font chữ
  - VALUES: normal | bold | bolder | lighter | inherit

Ví dụ: *font-weight: bold;*

Bold

- *font-variant*: tạo văn bản với dạng *small-cap style* – tất cả ký tự điều được viết hoa, ký tự đầu tiên lớn hơn so với các ký tự còn lại

VALUES: normal | small-caps | inherit

Ví dụ: *font-variant: small-caps;*

SMALL CAPS

3. Text

- Các thuộc tính text cho phép định dạng một đoạn văn bản, bao gồm: *color*, *letter-spacing*, *word-spacing*, *line-height*, *white-space*, *text-align*, *vertical-align*, *text-indent*, *text-decoration*, *text-transform*
- *color* : màu của văn bản
  - VALUES: <color> | inherit

  Ví dụ:

  color: rgb(128,0,0);

  text is maroon
- *letter-spacing* : khoảng cách giữa 2 ký tự trong văn bản
  - VALUES: <length> | <percentage-font-size>% | inherit

  Ví dụ:

  letter-spacing: .1em;

  letters are spaced apart

3. Text (tt)

- *word-spacing* : khoảng cách giữa 2 từ trong văn bản
  - VALUES: normal | <length> | inherit

Ví dụ: *word-spacing: 40px;*

words are spaced apart

- *line-height* : khoảng cách giữa các dòng trong một đoạn văn bản
  - VALUES: normal | <number> | <length> | <percentage-font-size>% | inherit

Ví dụ: *line-height: 2;*

One thing was certain, that the white kitten had had nothing to do with it: -- it was the black kitten's fault entirely.

1

One thing was certain, that the white kitten had had nothing to do with it: -- it was the black kitten's fault entirely.

1.5

One thing was certain, that the white kitten had had nothing to do with it: -- it was the black kitten's fault entirely.

2

3. Text (tt)

- *white-space* : xác định các ký tự white-space liên tiếp nhau có được giữ lại hay văn bản có thể gói lại (wrap) hay không
  - VALUES: normal | pre | nowrap | pre-wrap | pre-line | inherit

Ví dụ:

white-space: pre;

multiple spaces are not collapsed

- *text-align* : canh lề cho văn bản
  - VALUES: left | center | right | justify | inherit

Ví dụ:

text-align: center;

One thing was certain, that the white kitten had had nothing to do with it: -- it was the black kitten's fault entirely.

Left

One thing was certain, that the white kitten had had nothing to do with it: -- it was the black kitten's fault entirely.

Center

One thing was certain, that the white kitten had had nothing to do with it: -- it was the black kitten's fault entirely.

Right

One thing was certain, that the white kitten had had nothing to do with it: -- it was the black kitten's fault entirely.

Justified

3. Text (tt)

- *vertical-align* : canh lền văn bản theo chiều thẳng đứng so với thành phần lân cận, thường dùng để tạo các đoạn văn bản dạng super-scripting hay sub-scripting, hay canh lề nội dung của môt ô trong bảng
  - VALUES: baseline | sub | super | top | text-top | middle | bottom | text-bottom | <percentage-line-height>% | <length> | inherit

Ví dụ: *vertical-align: sub;*

X^{super} X_{sub}

- *text-indent* : xác định khoảng cách lùi vào trong của dòng đầu tiên trong văn bản
  - VALUES: <length> | <percentage-width>% | inherit

Ví dụ: *text-indent: 1em;*

One thing was certain, that the white kitten had had nothing to do with it: -- it was the black kitten's fault entirely.

3. Text (tt)

- *text-decoration* : tạo đường kẻ gạch qua văn bản (line-through, underline, overline) hay tạo văn bản nhấp nháy (blink)
  - VALUES: none | line-through | underline | overline | blink | inherit

Ví dụ: *text-decoration: strike-through;*

~~One thing was certain, that the white kitten had had nothing to do with it: -- it was the black kitten's fault entirely.~~

Line-through

One thing was certain, that the white kitten had had nothing to do with it: -- it was the black kitten's fault entirely.

Underline

One thing was certain, that the white kitten had had nothing to do with it: -- it was the black kitten's fault entirely.

Overline

- *text-transform* : tạo văn bản in hoa (uppercase), thường (lowercase), hay viết hoa các ký tự đầu tiên (capitalize)
  - VALUES: lowercase | capitalize | uppercase | none | inherit

Ví dụ: *text-transform: uppercase;*

text case

Lowercase

Text Case

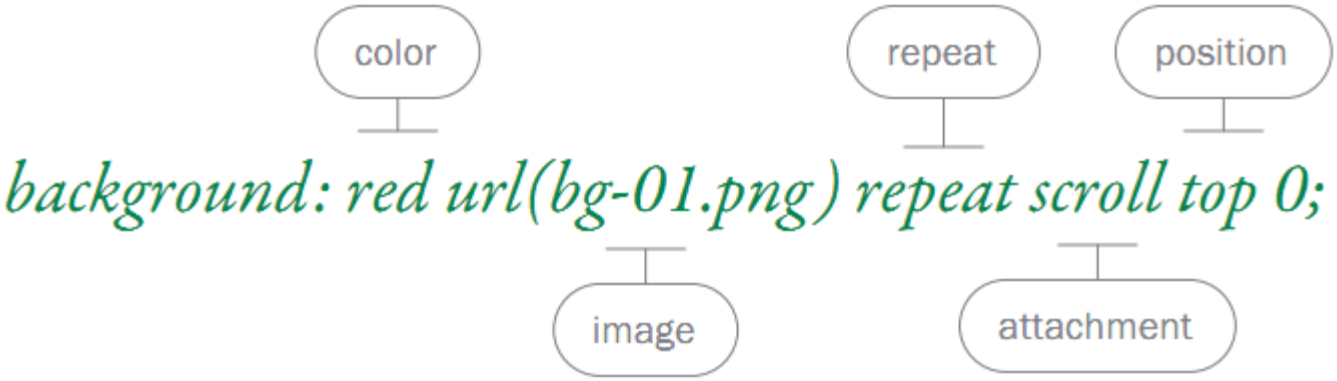
Capitalize

TEXT CASE

Uppercase

4. Background

- Hầu như tất cả các phần tử html điều có thuộc tính nền => dùng css để tùy chỉnh màu nền, ảnh nền
- background (shorthand property): cho phép thiết lập tất cả các thuộc tính nền của một phần tử
  - VALUES: <background-color> <background-image> <background-repeat> <background-attachment> <background-position> | none



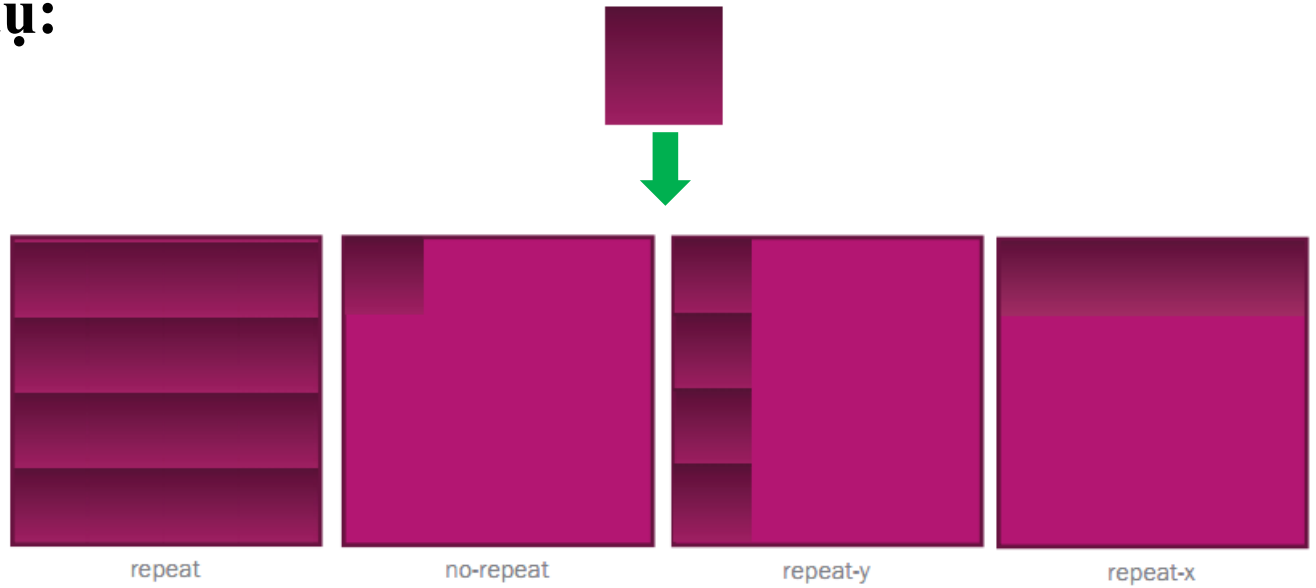
4. Background (tt)

- background-color: màu nền của một phần tử, nếu ảnh nền không phủ lắp hết phần tử thì phần còn lại sẽ được phủ lắp bằng màu nền
  - VALUES: <color> | transparent | inherit
- background-image: ảnh nền (png, jpeg, gif)
  - VALUES: url(<url>) | none
- background-attachment: thiết lập ảnh nền có được cuộn (scroll) theo nội dung hay cố định (fixed)
  - VALUES: scroll | fixed | inherit

4. Background (tt)

- background-repeat: thiết lập ảnh nền được lặp lại, lặp theo chiều ngang, chiều đứng hay không lặp. Mặc định ảnh nền sẽ được lặp lại để có thể phủ toàn bộ phần tử
  - VALUES: repeat | repeat-x | repeat-y | no-repeat | inherit

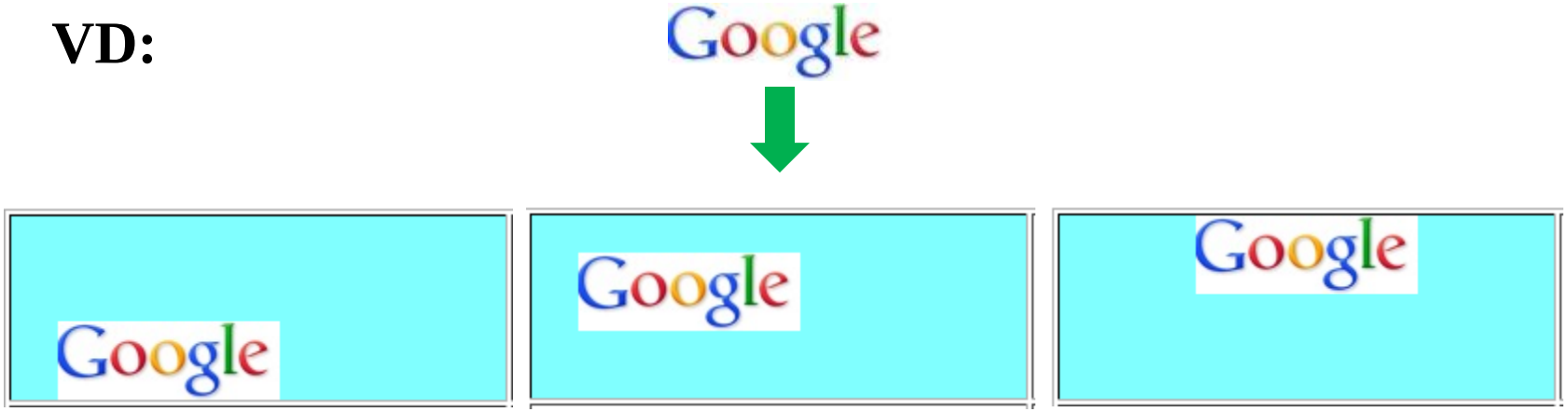
Ví dụ:



4. Background (tt)

- background-position: thiết lặp vị trí ảnh nền. Có thể thiết lặp cùng lúc vị trí ảnh nền ở cả 2 chiều ngang (left) và dọc (top) bằng cách đặc 2 giá trị cách nhau 1 khoảng trắng
  - VALUES: <length> | <percentage-box-width+padding>% | left | right | center | top | bottom | center | inherit

VD:



25px bottom | 25px 20px | top

5. Box model

- Tất cả phần tử HTML có thể được xem như các hộp (boxes). Trong CSS, thuật ngữ "mô hình hộp" (box model) được dùng khi nói về thiết kế và sắp xếp (layout).
- Mô hình hộp CSS là 1 hộp bao bọc các phần tử HTML, và chứa: lề (margins), đường viền (borders), vùng đệm (padding) và nội dung (content).
- Mô hình hộp cho phép thiết lặp các đường viền bao quanh các phần tử và không gian của các phẩn tử trong mối tương quan với các phần tử khác.

5. Box model (tt)

- **Lề (Margin):** chiếm một vùng quanh đường viền. Lề không có màu nền và nó hoàn toàn trong suốt.
- **Đường viền (Border):** Đường viền nằm giữa padding và nội dung. Đường viền bị ảnh hưởng bởi màu nền của hộp.

- **Padding:** chiếm một vùng quanh nội dung. Padding bị ảnh hưởng bởi màu nền của hộp
- **Nội dung (Content):** nội dung của hộp, nơi văn bản và hình ảnh xuất hiện

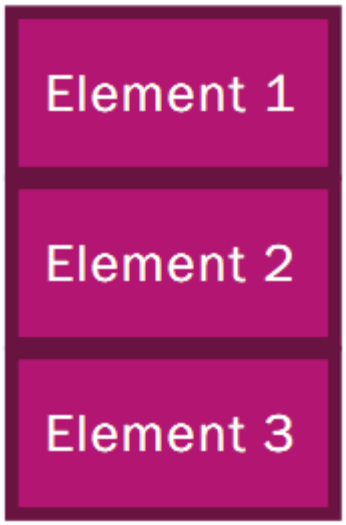
Display

- Tất cả các phần tử HTML sẽ được hiển thị theo một kiểu mặc định => dùng thuộc tính display để thiết lập cách hiển thị của một phần tử
- display: thiết lập phần tử HTML được hiển thị theo dạng inline, block, danh sách hay không hiển thị trên trình duyệt
  - VALUES: inline | block | list-item | none | inherit
  - Với giá trị none: phần tử HTML sẽ hoàn toàn bi gỡ bỏ khỏi trang web

Ví dụ:

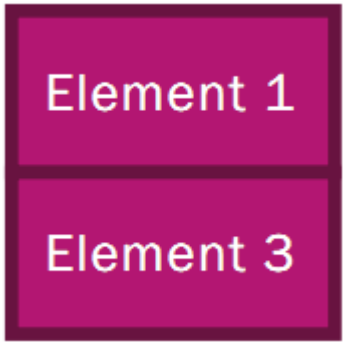
```
<span id="e1">Element 1</span>
<span id="e2">Element 2</span>
<span id="e3">Element 3</span>
```

Display (ví dụ)



#e1, #e2, #e3 { display: block; }

#e1, #e2, #e3 { display: inline; }



#e1, #e3 { display: block; }
#e2 { display: none; }

Visibility

- Visibility: hiển thị hay không hiển thị một phần tử html
  - VALUES: visible | hidden | inherit
  - Với giá trị là hidden phần tử sẽ không được trình duyệt hiển thị, tuy nhiên nó vẫn tồn tại trên trang web, và chiếm lấy khoảng không như lúc nó được hiển thị
- opacity: độ mờ đục của một phần tử từ 0.0 (trong suốt) đến 1.0 (hiển thị hoàn toàn)
  - VALUES: <0.0-1.0> | inherit
- filter: alpha (opacity=<0-100>): thiết lặp độ mờ đục của một phần tử từ 0 (trong suốt) đến 100 (hiển thị hoàn toàn). Đây không là thuộc tính trong chuẩn css, dùng thay thế cho thuộc tính opacity không được hỗ trợ trong IE 8 và các phiên bản trở về trước => đặt giá trị cho cả opacity & filter để có được kết quả giống nhau trên tất cả các trình duyệt.

Visibility (ví dụ)

```
<span id="e1">Element 1</span>
<span id="e2">Element 2</span>
<span id="e3">Element 3</span>
```

Element 1

Element 3

Element 1

Element 2

Element 3

```
#e1  { opacity: 1;
       filter: alpha(100); }
#e2  {  opacity: .5;
        filter: alpha(50); }
#e3 {  opacity: .25;
       filter: alpha(25); }
```

```
#e1, #e3 { display: block; }
#e2 { visibility: hidden; }
```

Float

- float: dùng để thiết lập vị trí của một phần tử bên trái, hay phải so với thành phần cha của nó. Tất cả các phần tử bên dưới nó sẽ "nổi" lên ngay bên cạnh nó chiếm lấy khoảng không gian còn trống.
  - VALUES: left | right | none | inherit
- clear: dùng để ngừng lại việc "nổi" lên của một phần tử
  - VALUES: none | left | right | both | inherit

Ví dụ:

```
<p id="e1">Element 1</p>
<p id="e2">One thing was…</p>
<p id="e3">he way Dinah…</p>
```

Float (ví dụ)

```
#e1{ width: 75px;
     height: 150px;
     float: right; }
#e3{ clear: right; }
```

One thing was certain, that the white kitten had had nothing to do with it: -- it was the black kitten's fault entirely. For the white kitten had been having its face washed by the old cat for the last quarter of an hour (and bearing it pretty well, considering); so you see that it couldn't have had any hand in the mischief.

Element 1

The way Dinah washed her children's faces was this: first she held the poor thing down by its ear with one paw, and...

```
#e1{  width: 75px;
      height: 150px;
      float: right; }
```

One thing was certain, that the white kitten had had nothing to do with it: -- it was the black kitten's fault entirely. For the white kitten had been having its face washed by the old cat for the last quarter of an hour (and bearing it pretty well, considering); so you see that it couldn't have had any hand in the mischief.

The way Dinah washed her children's faces was this: first she held the poor thing down by its ear with one paw, and...

Element 1

Width & height

- Thuộc tính width & height dùng để thiết lập độ rộng và chiều cho khu vực nội dung trong mô hình hộp
- width: độ rộng
  - VALUES: <length> | <percentage-parent-width>% | auto | inherit
- height: chiều cao
  - VALUES: <length> | <percentage-parent-height> | auto | inherit
- overflow: thiết lặp cách trình bày khi nội dung vượt quá sức chứa của một phần tử
  - VALUES: hidden | visible | scroll | auto | inherit

Width & height (tt)

VD:

width: 225px;

height: 150px;

overlow: hidden;

Border

- Các thuộc tính đường viền (border) cho phép định nghĩa các kiểu cho các đường viền xung quanh phần tử bao gồm: kiểu, màu sắc, độ rộng
- Có thể điều chỉnh kiểu đường viền cho từng phía khác nhau
- border (shorthand property): thiết lặp giá trị cho tất cả các thuộc tính của đường viền. Thứ tự các thuộc tính cần được đảm bảo như sau:
  - VALUES: <border-width> <border-style> <border-color>
- border-width: độ rộng đường viền
  - VALUES: <length> | thin | medium | thick | inherit
- border-color: màu sắc đường viền
  - VALUES: <color> | transparent | inherit

Border (tt)

- border-style: thiết lặp một kiểu được định nghĩa sẵn cho đường viền
  - VALUES: none | dotted | dashed | solid | double | groove | ridge | inset | outset | inherit
- border-top, border-right, border-bottom, border-left (shorthand properties):): thiết lặp giá trị cho tất cả các thuộc tính của đường viền theo từng phía
  - VALUES: <border-width> <border-color> <border-style>
- border-width-top, border-width-right, border-width-bottom, border-width-left , border-style-top, border-style-right, border-style-bottom, border-style-left , border-color-top, border-color-right, border-color-bottom, border-color-left: các thuộc tính dùng để điều chỉnh kiểu đường viền cho từng phía
  - VALUES: phụ thuộc vào từng loại thuộc tính cụ thể

Border (ví dụ)

Ví dụ: border: 5px solid rgb(67,0,37);

Top = 5px solid rgb(67,0,37)

Left = 5px solid rgb(67,0,37)

Right = 5px solid rgb(67,0,37)

One thing was certain, that the white kitten had had nothing to do with it: -- it was the black kitten's fault entirely. For the white kitten had been having its face washed by the old cat for the last quarter of an hour (and bearing it pretty well, considering); so you see that it couldn't have had any hand in the mischief.

The White Kitten

The way Dinah washed her children's faces was this: first she held the poor thing down by its ear with one paw, and then with the other paw she rubbed its face all over, the wrong way, beginning at the nose: and just now, as I said, she was hard at work on the white kitten, which was lying quite still and trying to purr -- no doubt feeling that it was all meant for its good.

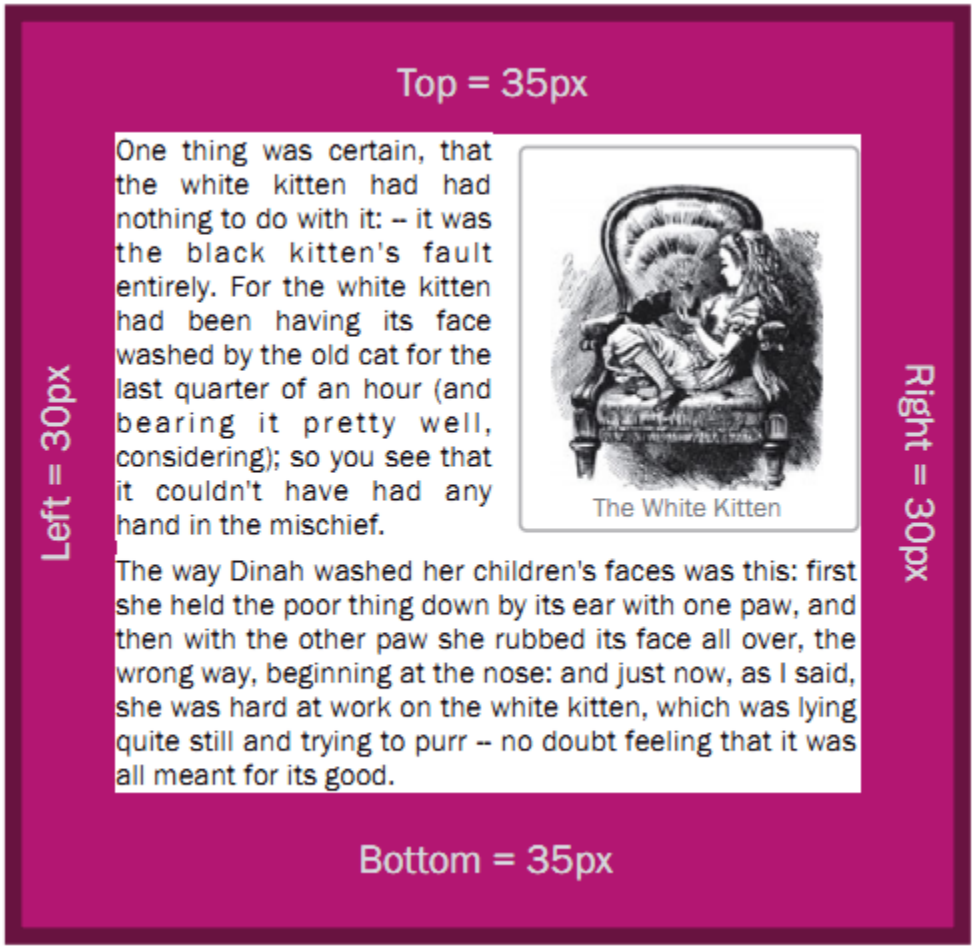
Bottom = 5px solid rgb(67,0,37)

Padding

- Padding (vùng đệm) là khoảng không giữa nội dung và đường viền của một phần tử
- Có thể điều chỉnh độ rộng vùng đệm cho từng phía khác nhau
- padding (shorthand property): thiết lặp độ rộng vùng đệm cho tất cả các phía
  - VALUES: <length> | <percentage-box-width>% | inherit
  - Thuộc tính padding có thẻ có từ 1 đến 4 giá trị
- padding-top, padding-right, padding-bottom, padding-left: thiết lặp độ rộng vùng đệm cho từng phía
  - VALUES: <length> | <percentage-box-width>% | inherit

Padding (ví dụ)

padding : 35px 30px;

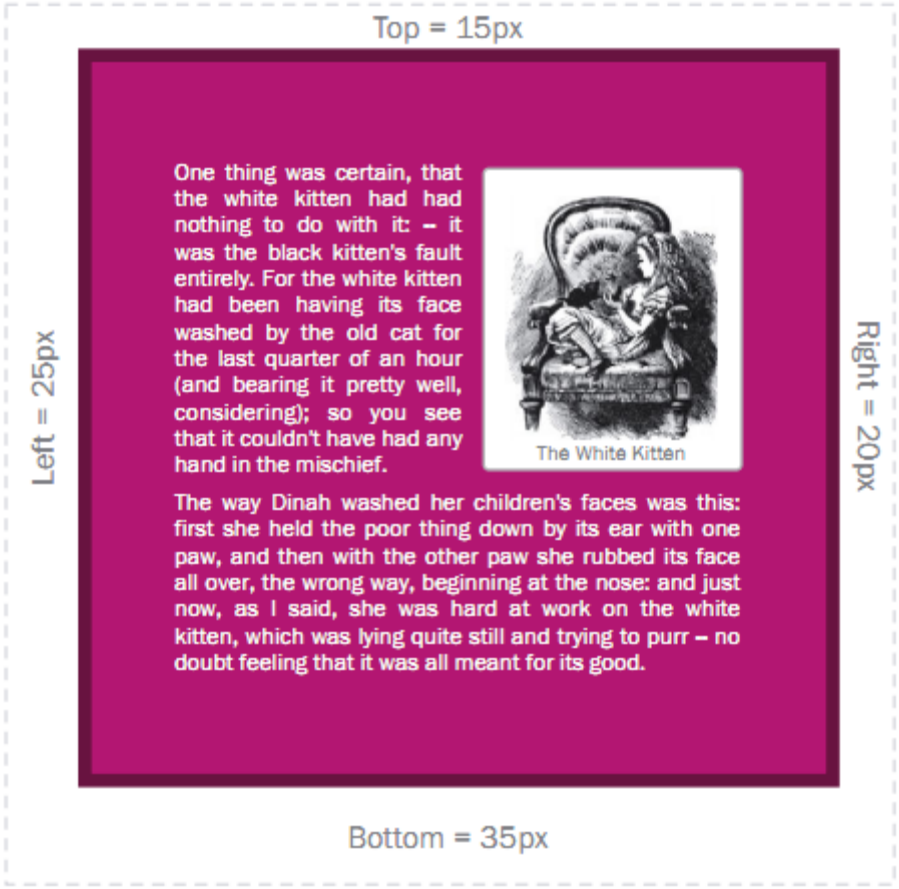


Margin

- Margin (lề): khoảng không giữa đường viền của một phần tử với đường viền của các phần tử lân cận
- Có thể điều chỉnh độ rộng đường viền cho từng phía khác nhau
- margin (shorthand property): độ rộng lề cho tất cả các phía
  - VALUES: <length> | <percentage-box-width>% |inherit
  - Thuộc tính margin có thể có tử 1 đến 4 giá trị
- margin-top, margin-right, margin-bottom, margin-left: độ rộng lề cho từng phía
  - VALUES: <length> | <percentage-box-width>% | inherit

Margin (ví dụ)

margin: 15px 20px 35px 25px;



Outline

- Outline là đường vẽ xung quanh các phần tử, ngoài rìa đường viền để làm nổi bật một phần tử
- Outline không ảnh hưởng đến kích thước của một phần tử
- Các thuộc tính outline xác định kiểu, màu, và độ rộng của outline, không thể điều chỉnh thuộc tính outline cho từng phía riêng biệt của một phần tử
- Outline (shorthand property): giá trị cho tất cả các thuộc tính outline
  - VALUES: <outline-width> <outline-color> <outline-style>
- outline-width: độ rộng của outline
  - VALUES: <length> | thin | medium | thick | inherit

Outline (tt)

- Outline-color: màu của outline
  - VALUES: <color> | invert | inherit
- Outline-style: thiết lập một kiểu được định nghĩa sẵn cho outline
  - VALUES: none | dotted | dashed | solid | double | groove | ridge | inset | outset | inherit

Ví dụ: outline: 10px solid rgb(67,0,37)

6. Position

- position: xác định cách định vị cho một phần tử
  - VALUES: static | relative | absolute | fixed | inherit
  - static: phần tử được định vị mặc nhiên, không thể thay đổi vị trí của phần tử (không bị ảnh hưởng bởi các thuộc tính top, bottom, left, right)
  - relative: vị trí của phần tử được tính tương đối so với vị trí mặc nhiên của nó, phần tử vẫn sẽ chiếm giữa khoảng không gian mà nó định vị như trước khi được di chuyển với các thuộc tính: top, bottom, left, right
  - absolute: vị trí của phần tử được tính tương đối so với phần tử cha đầu tiên của nó có giá trị của thuộc tính position khác static. Nếu không tồn tại phần tử như thế thì vị trí tương đối được tính so với phần tử <html>
  - fixed: vị trí của phần tử được tính dựa vào cửa sổ của trình duyệt, phần tử sẽ được cố định tại một vị trí khi màn hình được cuộn
- Một phần tử có thuộc tính position có giá trị: relative, absolute, fixed sẽ có khả năng phủ lắp lên một phần tử khác

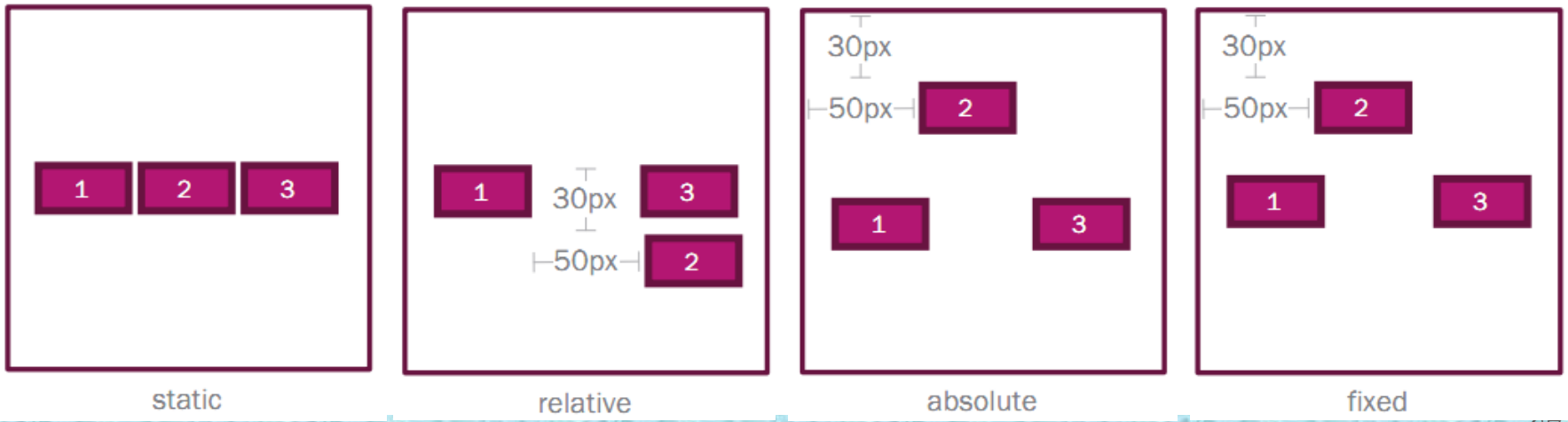
6. Position (tt)

- Thuộc tính position không làm thay đổi vị trí của một phần tử, nó chỉ là bước chuẩn bị để ta có thể định vị một phần tử thích hợp bằng các thuộc tính sau:
  - top: di chuyển phần tử một khoảng cách tính từ lề phía trên
    - VALUES: auto | <length> | <percentage-parents-height>% | inherit
  - right: di chuyển phần tử một khoảng cách tính từ lề phía bên phải
    - VALUES: auto | <length> | <percentage-parents-width> % | inherit
  - bottom: di chuyển phần tử một khoảng cách tính từ lề phía dưới
    - VALUES: auto | <length> | <percentage-parents-height>% | inherit
  - left: di chuyển phần tử một khoảng cách tính từ lề phía bên trái
    - VALUES: auto | <length> | <percentage-parents-width>% | inherit

6. Position (tt)

Ví dụ:

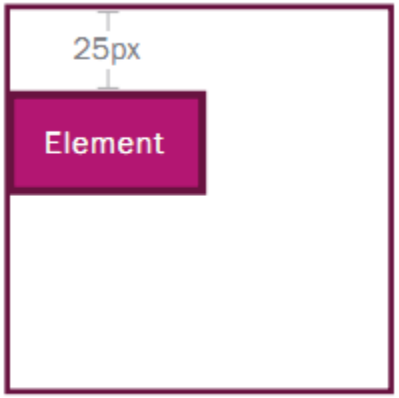
```
<span id="e1">1</span>
<span id="e2">2</span>
<span id="e3">3</span>
#e2{top:30px;
    left:50px;}
```



6. Position (tt)

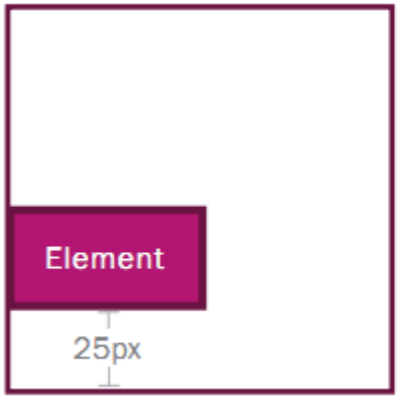
Ví dụ:

```
<span id="e2">Element</span>
#e2{ position: absolute; }
```

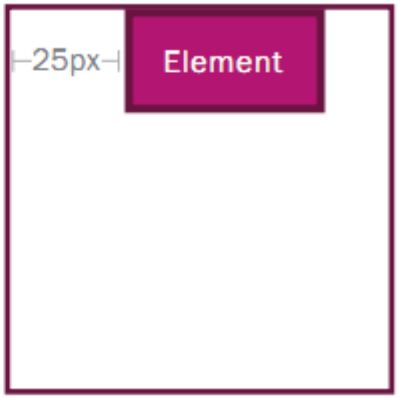


top: 25px

right: 25px



bottom: 25px



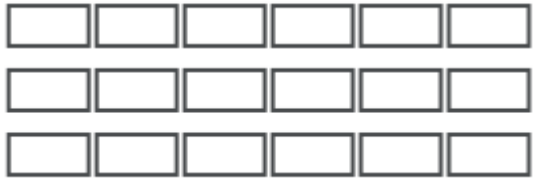
left: 25px

7. Tables

- border-spacing: khoảng cách giữa các ô trong bảng
  - V ALUES: <length> | inherit
- border-collapse: xác định đường viền giữa các ô kế nhau trong bảng có thể chia sẽ với nhau (collapse), hay không (separate)
  - VALUES: collapse | separate | inherit
- caption-side: vị trí tiêu đề của bảng
  - VALUES: top | bottom | inherit
- empty-cells: hiển thị hay không đường viền, nền của một ô dữ liệu trống trên bảng
  - VALUES: show | hide | inherit
- table-layout: lựa chọn giải thuật để trình bày một bảng
  - VALUES: auto | fixed | inherit

7. Tables (ví dụ)

border-spacing: 6px 2px;



border-collapse: collapse;

collapse

separate

caption-side: top;

top

bottom

8. Lists

- Dùng để tạo các kiểu đánh dấu cho các phần tử danh sách (<ul>, <ol>)
- list-style (shorthand property): giá trị cho tất cả các thuộc tính của danh sách
  - VALUES: inherit | <list-style-type> <list-style-image> <list-style-position>
- list-style-type: kiểu đánh dấu đầu dòng cho các phần tử trong danh sách
  - VALUES: disc | circle | square | decimal | decimal-leading-zero | upper-roman | lower-roman | upper-alpha | lower-alpha | lower-greek | none | inherit
- list-style-image: tạo kiểu đánh dấu đầu dòng cho các phần tử trong danh sách là một ảnh
  - VALUES: url(<url>) | none | inherit
- list-style-position: xác định lề, và vị trí đặt các ký tự đánh dấu đầu dòng của các phần tử trong danh sách
  - VALUES: inside | outside | inherit

8. Lists (vd)

list-style-type: square;

- Disc
- Circle
- Square

A Upper-Alpha

a Lower-Roman

α Lower-Greek

1 Decimal

01 Decimal-Leading-Zero

I Upper-Roman

i Lower-Roman

list-style-position: inside;

- One thing was certain, that the white kitten
- had had nothing to do with it
- it was the black kitten's fault entirely

Inside

- One thing was certain, that the white kitten
- had had nothing to do with it
- it was the black kitten's fault entirely

Outside

9. Cursor

- cursor: thiết lập hình dạng cho con trỏ
  - VALUES: default | url(<url>) | auto | crosshair | pointer | move | e-resize | ne-resize | nw-resize | n-resize | se-resize | sw-resize | s-resize | w-resize | text | wait | help | progress | inherit