# HTML

# 1. một số thẻ html cơ bản

**<!DOCTYPE html>**  => khai báo html5

**<html> </html>** => tất cả mã html phải được viết trong thẻ này

**<head> </head>** => chứa các khai báo và định nghĩa cho trang web

**<body> </body>** => chứa nội dung hiển thị của trang web

# 2. Cấu trúc cơ bản của 1 trang html

**<html>**

**<head>**

// code

**</head>**

**<body>**

// code

**</body>**

**</html>**

# 3. Thẻ meta (meta tags)

- được dùng để cung cấp thông tin cho trình duyệt về trang HTML

- thường là cách hiển thị trang web

một số ví dụ:

+ **<meta charset="UTF-8">** : đặt mã hoá ký tự cho trang web

+ **<meta name="viewport" content="width=device-witdh, initial-scale=1.0">**

# 4. Một số thẻ thường dùng trong body

**<p> </p>** : thể hiện 1 đoạn văn bản

+ chuỗi ký tự **&nbsp;** dược thay cho khoảng trắng nếu văn bản có nhiều khoảng trắng liền nhau.

+ **<br>** : sử dụng để xuống dòng (phím enter sẽ không có tác dụng xuống dòng)

+ thêm thuộc tính align để căn lề vd: **<p align=”value”>** Văn bản **</p>**

* Value có thể là: left, center, right, justify.
* Mặc định căn trái (left)

**<h1> </h1> :** thẻ tiêu đề (heading)

+ thẻ tiêu đề sẽ có font-size lớn và in đậm.

+ kích thước sẽ giảm từ **<h1>** đến **<h6>**

Một số thẻ định dạng văn bản:

**<b>** : in đậm

**<i>** : in nghiêng

**<u>** : gạch chân

**<s>** : gạch ngang

**<big>** : chữ to

**<br>** : xuống dòng

**<hr>** : đường kẻ ngăn dòng trên và dòng dưới

**<bdo>** : đảo ngược chữ

# 5. Chú thích (comment)

**<!--**nội dung chú thích**-->** : chú thích được đặt trong cặp thẻ.

# 6. Liên kết (link) <a>

**<a href=”url”>**Nội dung**</a>**

+ <a> : là anchor(neo)

+ Các nội dung trong thẻ a khi nhấp vào sẽ đưa đến liên kết url trỏ tới

+ **href**: Xác định địa chỉ URL mà liên kết sẽ chuyển đến.

+ **target**: Xác định nơi để mở tài liệu được liên kết. Các giá trị phổ biến bao gồm:

**\_blank**: Mở liên kết trong một cửa sổ hoặc tab mới.

**\_self**: Mở liên kết trong cùng một khung (mặc định).

**\_parent**: Mở liên kết trong khung cha của khung hiện tại.

**\_top**: Mở liên kết trong toàn bộ cửa sổ.

+ nếu href trỏ đến tài liệu (.docx, .txt, .html, ....) thì sẻ mở tài liệu đó

+ thêm thuộc tính download vào thẻ <a> để tải về tài liệu đó

VD : **<a href=”xinchao.txt” download>** //tải về file xinchao.txt

# 7. bookmark

+ Là 1 điểm đánh dấu vị trí trên trang.

**<h1 id=”bookmark1”></h1>** : dùng từ khoá **id=”Tên bookmark”** để đánh dấu bookmark

+ Di chuyển đến điểm đánh dấu bookmark

**<a href=”index.html#bookmark1”><a> :** dùng liên kết bình thường nhưng thêm #”Tên bookmark”

**<a href=”#bookmark1”><a>:** Khi dùng trong nội bộ trang web thì không cần url.

Mỗi bookmark cung cấp 1 định danh duy nhất cho một phần tử, phần tử khác không được đặt cùng tên đó.

Gọi id trong css : **#tenid {} :** những cấu hình trong css chỉ áp dụng cho element có id này

# 8. Thuộc tính (attribute)

Khi khai báo 1 thuộc tính cho phần tử ta viết vào bên trong **thẻ mở** của phần tử.

**<h1 attribute=”value”> nội dung thẻ <h1>**

Vd **<h1 align=”right”>ND<h1>** : thuộc tính align, giá trị right ,nghĩa là căn lề phải cho văn bản.

# 9. Thuộc tính style

**Là thuộc tính để áp dụng các định dạng css cho html**

Có 3 cách triển khai

1. **Triển khai trực tiếp trong thẻ**

**<h1 style=”font-size:50px; color:violet”> Nội dung <h1>**

-Đặt trong đấu “ ” và mỗi thuộc tính css cách nhau bởi dấu ;

**2. Triển khai trên thẻ head**

<head>

<style>

// áp dụng cho toàn bộ thẻ p trong file html

p{

Các thuộc tính css;

}

</style>

</head>

1. **Triển khai ở file style.css**

**<link rel=”stylesheet” href=”đường dẫn file style.css”>**

Thêm vị trí file style.css vào head sau đó định nghĩa các thuộc tính trên file style.css

# 10. id và class

Cả id và class đều dùng để đặt tên cho phần tử

* **id**: tên phải là **duy nhất**, không được trùng lặp **<a id=”idname”></a>**
* **class** : tên có thể trùng lặp **<a class=”classname”></a>**
* tên id và class không được chứa ký tự đặc biệt(ngoại trừ ’\_’ và ‘-‘<) và không được bắt đầu bằng số
* có thể dùng tên id để đặt cho class
* tên id và class có phân biệt hoa thường
* một element có thể có nhiều class
* để gọi **id:** **#”idname”**
* để gọi **class:**  **.”classname”**

# 11. phần tử khối, phần tử nội tuyến

|  |  |
| --- | --- |
| **Phần tử khối** | **Phần tử nội tuyến** |
| - Luôn được bắt đầu ở 1 hàng mới.  - Chiều rộng luôn chiếm toàn bộ phần nội dung | - Phần tử hiển thị nối tiếp phần tử tiếp  - Chiều rộng của phần tử phụ thuộc vào nội dung bên trong nó.  - |

<b> </b> : bold ký tự

<s> </s> : gạch ngang ký tự

<u> </u> : gạch chân ký tự

<i> </i> : in nghiêng ký tự

<em></em> : emphasis nhấn mạnh văn bản (bằng in nghiêng)

Để kiểm tra 1 phần tử là khối hay nội tuyến thì thêm đường viền cho nó.

# 12. thẻ <div>, thẻ <span> (div là 1 thẻ quan trọng và thường dùng)

<div> divison (phân chia) : là phần tử khối (block) dùng để chia hoặc nhóm các phần tử thành khối, để sắp xếp và tổ chức nội dung hợp lý.

Dùng thẻ <div> sẽ tạo ra một khối trắng, thêm các attribute để định dạng nó.

Các attribute sẽ ảnh hưởng đến tất cả các thành phần bên trong nó.

<span> (bao phủ, trải dài) : giống thẻ div nhưng là phần tử nội tuyến(inline)

Thường dùng để xử lý 1 đoạn văn bản mà không ảnh hưởng đến các đoạn văn bản khác.

# 13. Chèn hình ảnh, thẻ <img>

**<img src=”imageUrl”> :** chèn hình ảnh vào.

Một số attribute để định dạng cho img

alt=”text” : thay thế 1 đoạn text khi img ko load được.

width=”value” & height=”value” : chỉ định chiều rộng và chiều cao của ảnh (ảnh sẽ bị méo)

title=”text” : thêm một chú thích text khi di chuột qua.

align=”value”: set vị trí hình ảnh so với văn bản.

Value: left, right, top, middle, bottom

Border=”value”: tạo một đường viền xung quanh img

# 14. Danh sách (UL, LI)

<ul> :unordered list.

<li> :list item.

# 15. Bảng <table>

<table>

<thead> chứa các đề mục của bảng

<th> Mỗi thẻ này chứa một đề mục </th>

<th> Mỗi thẻ này chứa một đề mục </th>

</thead>

<tbody> chứa các phần tử con của bảng

<tr> (table row)chứa dữ liệu của 1 hàng

<td>Mỗi thẻ này chứa 1 phần tử con tương ứng với đề mục</td>

<td>số thẻ <td> phải tương đương thẻ <th> </td>

</tr>

</tbody>

</table>

# 16. input <input>

<input type=”text”>

Một số input type thường gặp

**Text** : nhập dữ liệu text

**Password** : thay text bằng ký tự \*\*\*\*

**Mail**: nhập mail có kiểm tra định dạng

**Number** : nhập số (dùng min=”” max=”” để giới hạn giá trị)

**Date**: ngày tháng

**Checkbox**: Chọn nhiều mục

**Radio** : chọn 1 trong các lựa chọn (lưu ý để name=”” giống nhau)

**File**: nhập file từ máy tính

**Submit**: gửi form

**Color**: chọn màu sắc

**Range**: dùng thanh trượt để chọn giá trị (dùng min=””, max=”” để giới hạn giá trị)

**Button**: Tạo một nút có thể tuỳ chỉnh bằng javascript

**Select**: thẻ này không nằm trong thẻ <input>, tạo ra danh sách chọn

<select name=””>

<option value=””> mô tả </option>

<option value=””> mô tả </option>

</select>

**Textarea**: thẻ này không nằm trong thẻ <input>, nhập text trong nhiều dòng

<textarea>Văn bản có sẵn</textarea>

# CSS

# Các cách dùng css

1. Internal – dùng thẻ <style></style> đặt trong html
2. External – dùng file style.css ở bên ngoài  
   import đường dẫn : <link rel=”stylesheet” href=”link đến file css”>
3. Dùng trực tiếp trên title dùng attribute <h1 **style=”thuộc tính 1; thuộc tính 2; thuộc tính 3”**></h1>

# ID & CLASS (css selector)

Cách để gọi đến các element.

Id: là định danh duy nhất   
Class : có thể trùng name

id=”idName” == > select id **#idName**  
class=”className” ==> select class **.className**

# Mức độ ưu tiên

Giữa internal và external không có ưu tiên.  
Mức độ ưu tiên

1. Tag (gọi thẳng tên thẻ khi css)
2. Class
3. Id
4. Inline

Mức độ ưu tiên 4 < 3 < 2 < 1

Equal specificity : khi gọi bằng 1 phương thức giống nhau thì cái nào gọi sau sẽ được ưu tiên  
có thể dùng nhiều cách kết hợp để select đối tượng nhằm chọn ra đối tượng phù hợp  
vd1: h1.className#idName {} : chọn đối tượng thẻ h1 có class là className và id là idName.  
vd2: h1.className{} : chọn đối tượng thẻ h1 có class là className

Cách tính mức độ ưu tiên:  
vd1: 1 + 2 + 3 = 6  
vd2: 1 + 2 = 3  
Mức độ ưu tiên của vd1 sẽ cao hơn của vd2 do vậy vd1 sẽ luôn được hiển thị dù đặt trước hay sau vd2

Universal selector and inherited  
Universal : khi dùng thẻ \* thì tất cả các đối tượng html đều được css tuy nhiên ưu tiên chỉ ở mức 0  
Inherited: Các thẻ con sẽ kế thừa các thuộc tính css của thẻ cha.

!important : đặt sau các thuộc tính css (vd: color: blue !important)  
khi có từ khoá !important phía sau, các thuộc tính sẽ luôn được ưu tiên cao nhất  
khi có 2 !important cho cùng 1 đối tượng thì lại quay lại xét độ ưu tiên

# Biến (Variable)

Đặt trong lớp giả

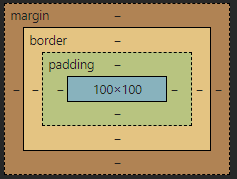
:root{ đặt biến kiểu global  
 --text-color: blue; tên biến : kiểu dữ liệu  
}

H1{ đặt biến kiểu local  
 --text-color:green;  
 color: **var**(--text-color); dùng biến phải dùng từ khoá var  
}

# Đơn vị (Unit)

1. Đơn vị tuyệt đối (**absolute**)
   * px
2. Đơn vị tương đối (**relative**)
   * % : phụ thuộc vào thẻ chứa nó
   * rem : phụ thuộc vào thuộc tính font-size của thẻ html (1 rem = 1 đơn vị font-size)(mặc định 16px)
   * em : phụ thuộc vào thẻ chứa nó gần nhất có thuộc tính font-size (1em bằng 1 đơn vị font-size)
   * vw : (viewport-width) phụ thuộc vào viewport (kích thước trình duyệt)
   * vh : (viewport-height) phục thuộc vào viewport

# Padding, Border, Margin



* + Cách viết cho cả 3
    - Padding : 10px (cả 4 cạnh)
    - Padding : 10px 10px (trên dưới, trái phải)
    - Padding : 10px 10px 10px 10px (trên trái dưới phải)
  + **Padding** : đệm thêm bằng chính đối tượng đó (dùng chính backgroud-color)
  + **Border** : thêm khung cho đối tượng (mặc định màu trắng)
    - Border có thêm thuộc tính border-color và border-style
    - Border : 10px solid #3333 (viền 10px , kiểu solid, mã màu)
  + **Magine** : thêm khoảng cách cho đối tượng đối với các đối tượng khác.

Padding và border làm tăng kích thước đối tượng.

Magine không làm tăng kích thước đối tượng.

# BOX-SIZING

Thuộc tính của các đối tượng.

Một số thuộc tính

* + content-box : giá trị mặc định, kích thước của đối tượng chỉ phụ thuộc vào nội dung của nó (nội dung là các thuộc tính được định nghĩa cho nó như là font-size hay width height, không phải là nội dung nó chứa)
  + border-box : kích thước của đối tượng sẽ không đổi khi dùng border hay padding.

# Background

-**Background-color** (tạo màu cho background)

-**Background-clip** (đi theo background color)  
 Xác định phạm vi của 1 phần tử sẽ được vẽ (có chứa padding hay border hay không)

* + - Border-box : vẽ từ border vào (mặc định)
    - Padding-box : vẽ từ padding vào
    - Content-box : không vẽ từ cả border và padding, chỉ vẽ bên trong phần nội dung

-**Background-image** (tạo background là 1 img, có thể dùng nhiều img đè nhau)

-**Background-size**: xác định kích thước của img (dùng đơn vị hoặc từ khoá)  
 Một số từ khoá: contain (lấy chiều dài nhất của bức ảnh mà không bị che khuất)  
 cover (lấy cạnh dài nhất lấy 100% khung nhìn chấp nhận che khuất)

-**Background-origin** (giống background-clip nhưng dùng cho background-img)

-**Background-position** (căn vị trí cho hình ảnh) (dùng đơn vị hoặc từ khoá)

Một sốt từ khoá: top, left, bottom, right, center  
 kết hợp 2 từ khoá để có vị trí mong muốn  
 Khi dùng 1 từ khoá, từ khoá phía sau sẽ mặc định là center

Syntax rút gọn

**Background**: url() / contain top  
 Dùng từ khoá **background:** và các tham số để set nhanh các tham số cho background

# Hàm trong CSS (Function)

* + Rgb
  + Cal
  + Attr

# Lớp giả (pseodo class)

Syntax để gọi pseodo class  
vd: h1**:hover**{}

Một số pseodo class thường gặp

:hover (xử lý khi user trỏ chuột (hover) vào đối tượng)  
:active(xử lý khi user click giữ vào đối tượng)  
:first-child (xử lý phần tử con đầu tiên)  
:last-child (xử lý phần tử con cuối cùng)  
:n-child (xử lý phần tử con thứ n)

# Phần tử giả (pseodo element)

Syntax để gọi pseodo element  
Vd h1::before{}

Một số pseodo element thường gặp

::before (thêm nội dung vào phía trước 1 phần tử)  
::after (thêm nội dung vào phía sau 1 phần tử)  
::first-line (áp dụng hiệu ứng cho dòng đầu tiên của văn bản)  
::first-letter (áp dụng hiệu ứng cho chữ cái đầu tiên của văn bản)  
::selection (áp dụng hiệu ứng cho đổi tượng được chọn (bôi đen))

# Position

Dùng để điều chỉnh vị trí của đối tượng, đối tượng có thể đè lên nhau

Relative : lấy chính nó làm gốc toạ độ  
Absolute : lấy thẻ cha gần nhất có chứa position làm gốc toạ độ  
Fixed : lấy cửa sổ trình duyệt làm gốc toạ độ, luôn nằm trên 1 vị trí tại cửa sổ trình duyệt)  
Sticky : bám dính trên cửa sổ trình duyệt khi cuộn ra ngoài trình duyệt

Top, left, right, bottom (dùng để điều chỉnh vị trí cho đối tượng đã được khai báo position)

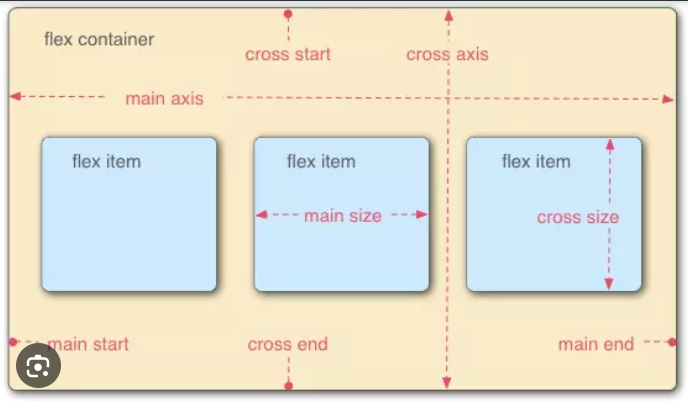
# Flex

Flex là giá trị (value) của thuộc tính display (display atttibute)  
display:flex;

Khi khai báo display:flex thì đối tượng sẽ trở thành flex-container và các phần tử con của nó  
sẽ trở thành flex-items.

Một số attribute phổ biến của flex

**Justify-content**: điều chỉnh các flex items theo chiều ngang.  
**align-items**: điều chỉnh các flex-items theo chiều dọc.  
**flex-direction**: xác định hướng sắp xếp của các flex-items (theo chiều dọc, ngang hoặc ngược lại)  
**flex-wrap**: cho phép các items xuống dòng hay không xuống dòng nếu chúng không vừa 1 dòng  
**align-self**: điều chỉnh căn chỉnh cho 1 items cụ thể (viết trong css của item)  
**order**: xác định thứ tự của item (viết trong css của item)



Các chiều của css được quy định theo hình trên (gồm start và end)   
**main axis** hiểu như là phương (nằm ngang, nằm dọc)  
**cross axis** luôn vuông góc với main axis  
**main start** và **main end** hiểu như là chiều (trái -> phải ....), sẽ quyết định sắp xếp items như thế nào  
**cross start** và **cross end** giống main nhưng là chiều trên dưới