HỌC HTML VÀ CSS – F8

Đặt font-size cho web:

html {

    font-size: 62.5%;

}

#heading {

    font-size: 2rem;

}

.sub-heading {

    font-size: 1.6rem;

}

.text, ol li {

    font-size: 1.4rem;

}

1. h1 - h6 - Heading
2. title – name of web page
3. p - paragraph
4. img - image – ảnh: src: link nguồn, alt=”tên khi bị lỗi”
5. a - anchor – neo link á
6. ul, ol, li – unordered list; ordered list; list
7. table bên trong có

<thead>

<th> </th>

<th> </th>

<th> </th>

</thead>

<tbody>

<tr>

<td> </td>

<td> </td>

<td> </td>

</tr>

tabble row,

**bên trên có bao nhiêu th thì bên dưới có bấy nhiêu tr**

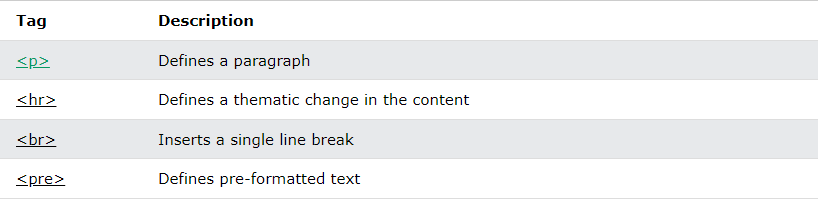
</tbody>

1. **input** :có type và name, type có 3 loại là

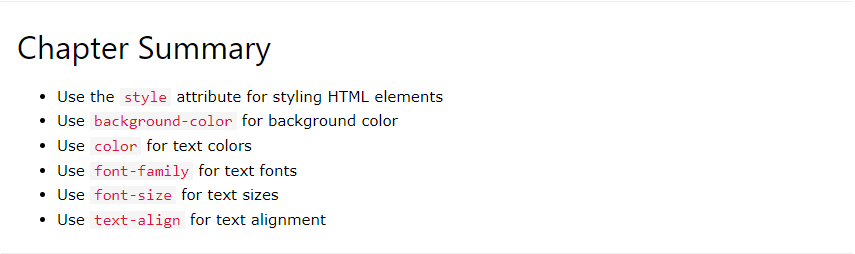
* text: khung nhập văn bản
* checkbox: ô vuông đánh tick
* radio: ô tròn đánh tick (khi trùng name, sẽ chỉ chọn được một ô)

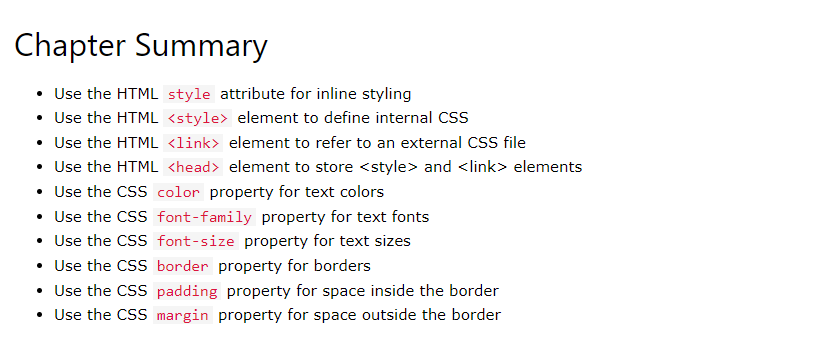
1. botton
2. div

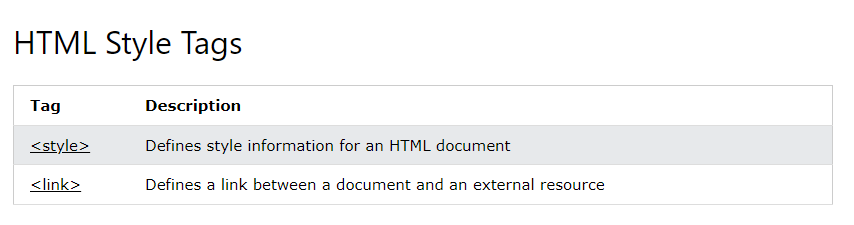
* Ghi chú trong html (comments): cú pháp: <!-- --> : phím tắt **Ctrol + /**
  + Sử dụng để ghi chú hoặc vô hiệu hóa code
* Thuộc tính attribute
  + <h1 **title = “đây là tiêu đề**> Heading </h1> tô đậm chính là attribute, thẻ title đó chính là thuộc tính trong thẻ mở html
* <hr> đường kẻ ngang, ngăn cách các đoạn nội dung.
* <pre> giữ nguyên I chang văn bản bên trong, có font mặc định, ko bị gom chung như thẻ <p>  
   </pre>

****

**HTML style**



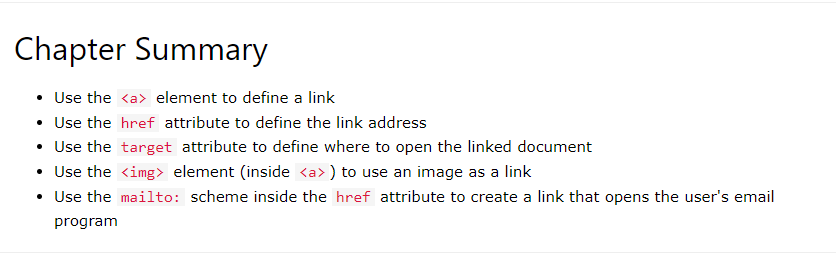
 





IMG as a link

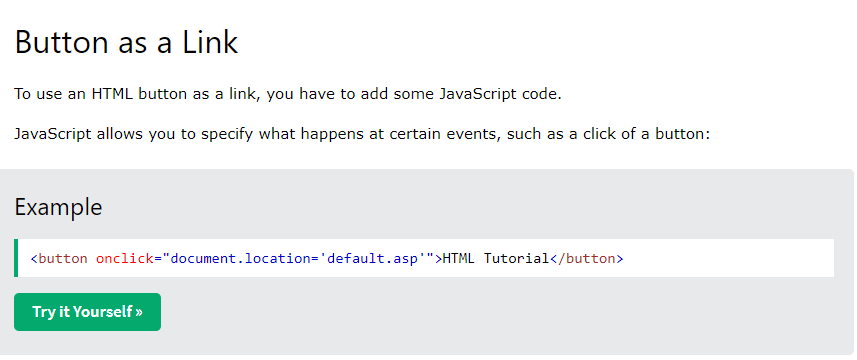
HTML link

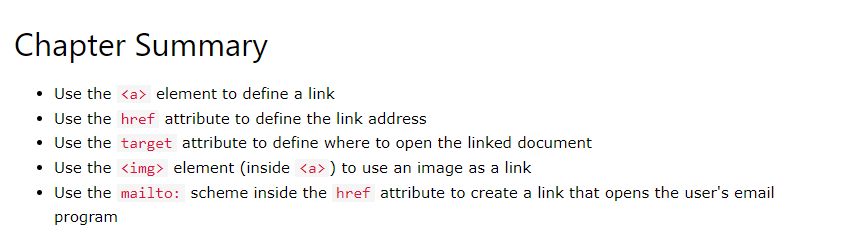


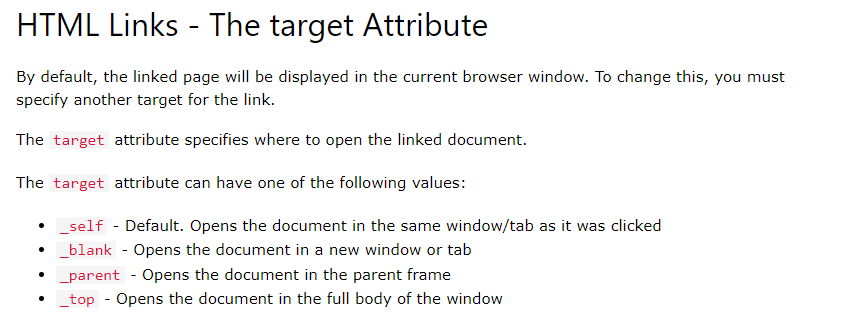




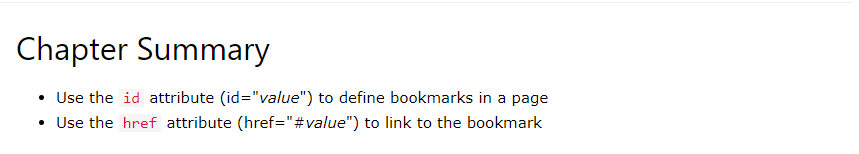
<a href=<mailto:silvergun2809@gmail.com>>Send email</a>







Tạo bookmark



HTML Image

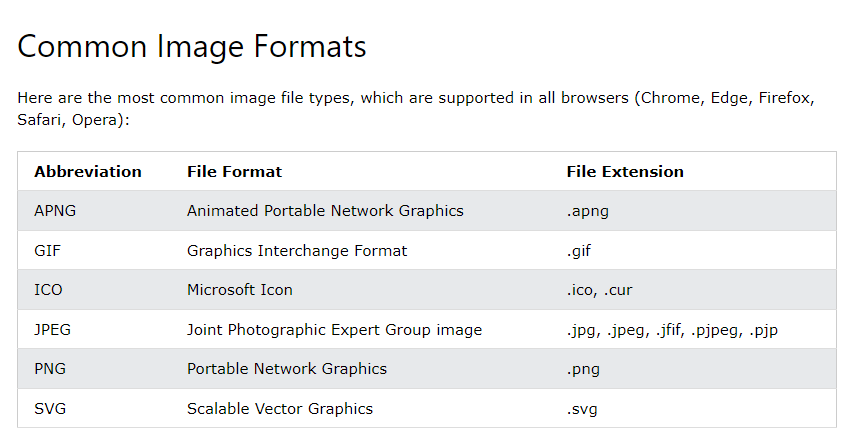
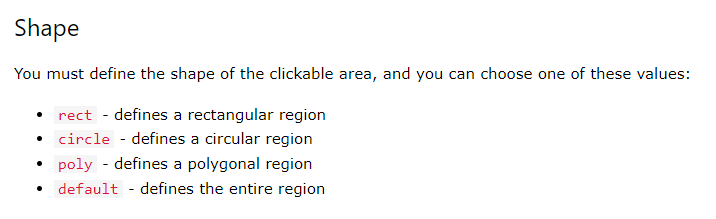
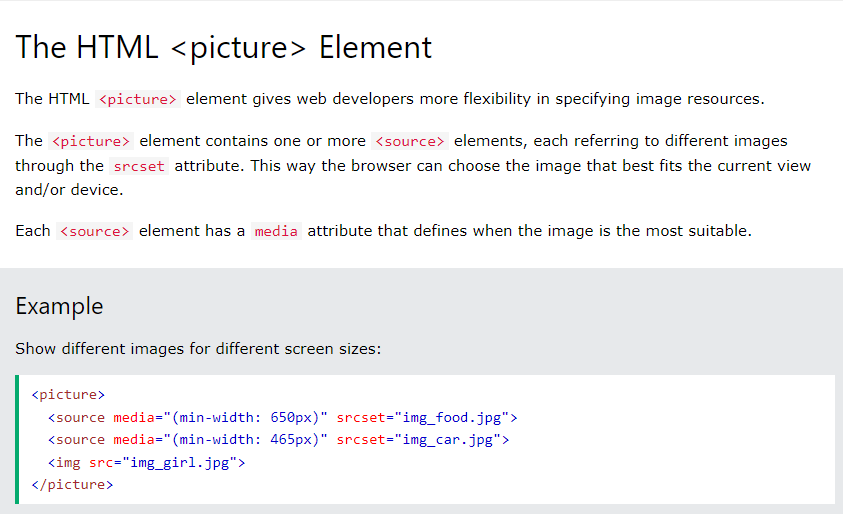


IMAGE MAP



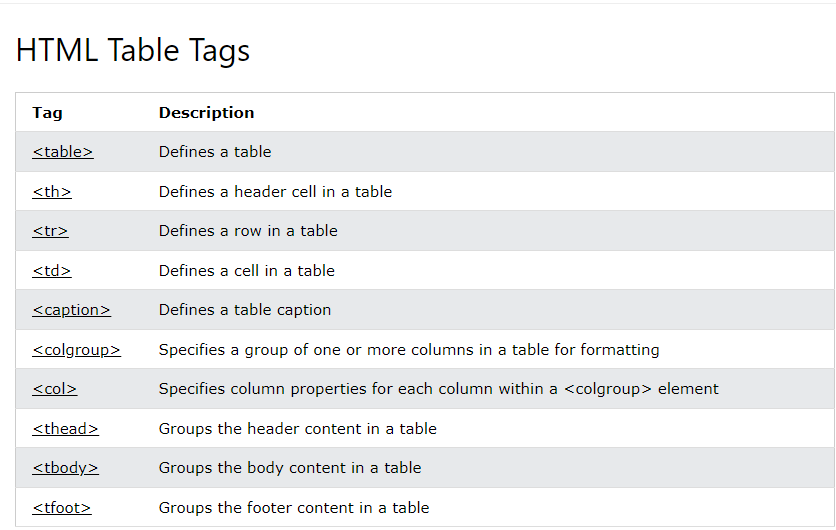
THE PICTURE ELEMENT



HTML Favicon – a small icon displayed next to the page title in the browser tab.

<link *rel*="icon" *type*="image/x-icon" *href*="/PJ/favicon.jpg">

HTML Table



Tabble style

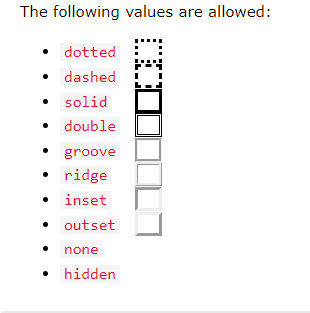
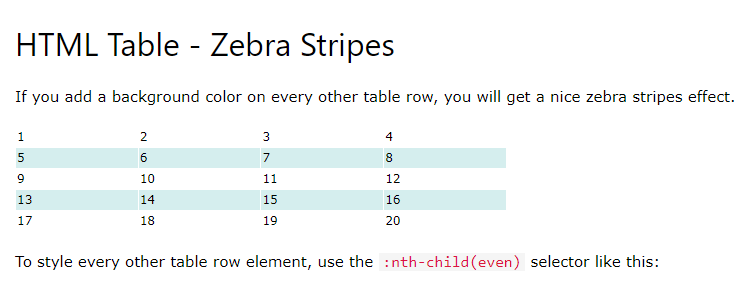
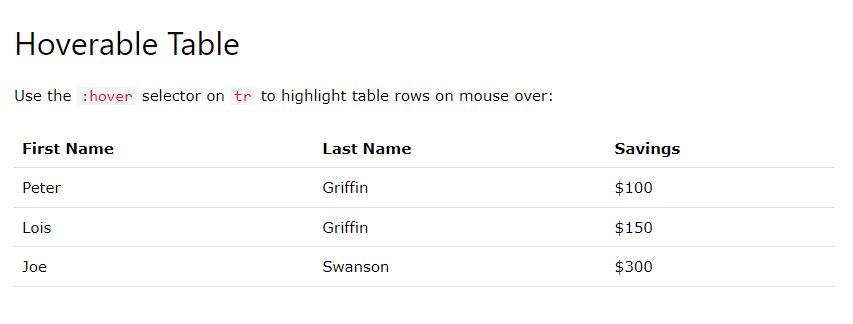


Table zebra stripes





HTML list

**HTML iframes:** An HTML iframe is used to display a web page within a web page.

<iframe src="demo\_iframe.htm" height="200" width="300" title="Iframe Example"></iframe>

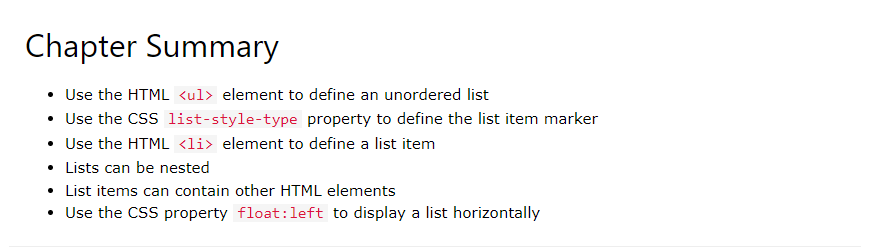
* Remove border by style inline or in CSS

<iframe src="demo\_iframe.htm" style="border:none;" title="Iframe Example"></iframe>

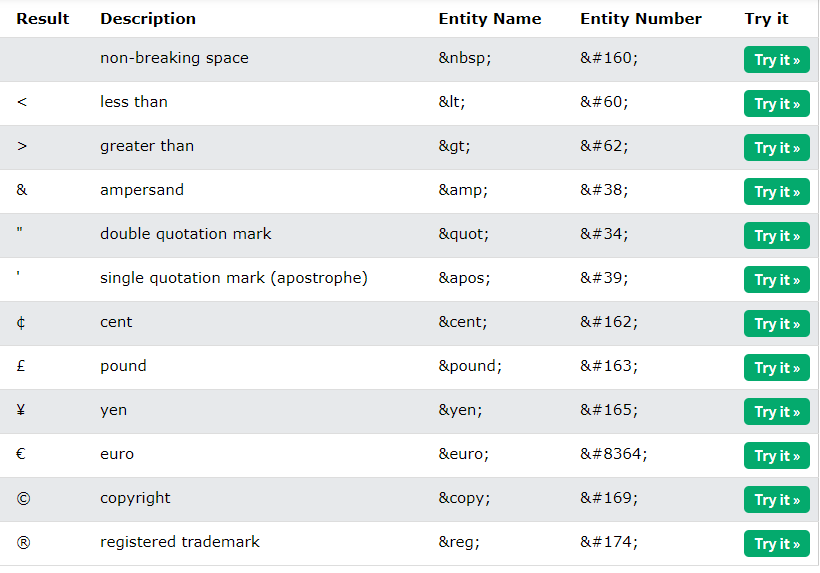
* Iframe – target for a link: An iframe can be used as the target frame for a link.

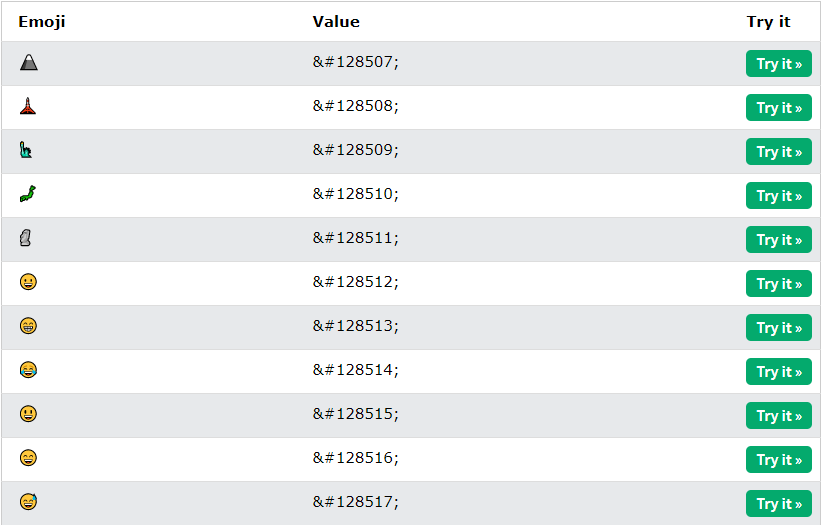
<iframe src="demo\_iframe.htm" name="iframe\_a" title="Iframe Example"></iframe>  
  
<p><a href="https://www.w3schools.com" target="iframe\_a">W3Schools.com</a></p>

Khi ấn vào link, target sẽ gọi web w3 trong cái iframe đang hiển thị web khác.



* Thẻ <address> cho địa chỉ.
* <cite> cho tác phẩm nghệ thuật.
* <bdo dir=”rtl”> để viết ngược từ phải sang trái.



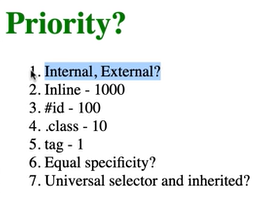


bài 10: ID và Class

Css selector: chọn một cái trong html bên css rồi css thuộc tính cho nó đặt id sau thẻ. vd <h1 id ="first heading"> sau đó gọi nó bên css: #first heading và cho thuộc tính của nó Tại sao phải làm như vậy? tại vì khi trong file html của mình có quá nhiều thẻ h1 mà mình không biết css vào thẻ nào, hoặc là mình muốn css vào một thẻ h1 này thôi thì id sẽ giúp mình phân biệt thông qua đặt tên id. tại sao không đặt đc 2 id giống nhau bởi vì tính định danh, sau này khi mà mình học sang Javascript thì nó bị lỗi.

Còn class kiểu như mình sẽ định danh ai là một lớp, và khi css thì mình gọi lớp đó sau dấu chấm và css một lần được nhiều đối tượng giống nhau

Độ ưu tiên trong Css: 1. Internal & External Không có cách nào là ưu tiên cả, cách nào được gọi sau sẽ thực hiện sau, cái nào mới hơn sẽ được thực hiện thôi. 2. Inline - 1000, #id - 100, .class - 10, tag - 1 Có nghĩa là ông nào nhiều điểm hơn thì sẽ được ưu tiên hơn. Khi sử dụng chỉ có id thì id nào được gọi sau sẽ được thực hiện Còn khi một element có cả id và class thì có thể gọi id và cả class bằng cú pháp #heading-id.heading-class thì cú pháp này sẽ được 110 điểm và do đó cũng win so với id



**LỚP GIẢ TRONG CSS**

::root {

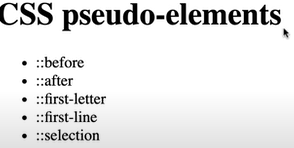
--text-color:red;

}



* :root – kiểu lớp giả để tạo biến css, sd để lấy màu
* :hover – kiểu lớp giả mà css xong, sau đó khi mình rê chuột vào thì nó mới hiện cái hiệu ứng đó còn không thì bình thường không hiện.
* :active – sử dụng để khi mình ấn vào thì nó đổi màu chữ, sử dụng để thay đổi màu của link…
* :first-child, :last-child – sử dụng để css vào con đầu tiên hoặc con cuối cùng của một thẻ cha.

**PHẦN TỬ GIẢ TRONG CSS**

****

* ::before, ::after – trong một phần tử div, mình có thể tạo ra một khối, hoặc dùng để viết mà không phải tạo ra một phần tử div khác nằm trong thẻ div ban đầu, và có thể css bình thường.
* ::first-letter – sử dụng để css cho phần tử đầu tiên. :: first-letter thường được sử dụng trong các bài báo cũ khi mà chữ cái đầu của bài viết thường được viết to hơn so với những chũ cái khác   
  Ví dụ: *h1::first-letter { font-size: 50px; }*
* ::first-line – sử dụng để css cho dòng đầu tiên của h1, p …
* ::selection là khi mình chọn vào nó ( bôi đen ) thì nó sẽ hiện lên các thuộc tính mà mình css trong selection vd h1::selection { background-color: violet; font-size: ...} thì khi mình bôi đen nó sẽ hiện lên các thuộc tính của selection.

Khi bỏ đối tượng được chọn đi vd như bỏ h1 chỉ còn lại ::selection thì all các element khi được chọn chúng sẽ thể hiện thuộc tính selection

Sử dụng biến để thay đổi màu, khi cần thay đổi số lượng lớn màu không cần thay đổi từng dòng.

**CLASS – ID – TAG**

Trường hợp mà mình muốn gọi một element có cả tag, id và class thì cho id về đầu: h1#heading-id.heading-class

Universal selector & inherited: Trường hợp mình muốn all element thừa hưởng cùng một yếu tố, vd như tất cả các element màu đen thì dùng cú pháp \* { color: } hoặc html { color: } và 2 yếu tố này đều 0 điểm

!IMPORTANT : đó là khi mình sử dụng thẻ tag và dùng thêm ! important thì nó sẽ win các ông còn lại kể cả inline, còn nếu tất cả các thẻ đều có ! important thì lại quay về xét điểm thoiy

BÀI 12: ĐẶT BIẾN TRONG CSS - Sử dụng một lớp giả :root { --varible-name: color ; } sau đó khi css cho heading chẳng hạn thì mình sẽ sử dụng một hàm var( --varible-name ); để sau này khi mình đặt màu cho nhiều element, thì khi mình có nhu cần sửa màu, chỉ cần sửa trên biến global trong lớp giả root mà thôi, còn biến local sẽ được đặt bên trong thẻ h1 khi mình gọi nó luôn.

13. CÁC ĐƠN VỊ TRONG CSS - CSS UNITS

1. Đơn vị tuyệt đối - Absolute units cm, mm, inch, px, pt, pc

2. Đơn vị tương đối %, rem, em, vh, vw.

Đơn vị tuyệt đối là khi những yếu tố xung quanh thay đổi thì bản thân nó sẽ không thay đổi, còn đơn vị tương đối thì ngược lại. Các đơn vị của giá trị tương đối thì nó cần một cái để phụ thuộc vào vd như là thẻ body, nếu dùng phần trăm thì nó sẽ là bao nhiêu phần trăm của thẻ body đấy. % sẽ phụ thuộc vào thẻ chứa nó rem sẽ phụ thuộc vào thẻ html. Vd css cho thẻ h1 là 1 rem, còn của thẻ html mình css cho nó 40px thì khi refresh cỡ chữ của h1 sẽ là 40px, nói chung rem nó sẽ phụ thuộc vào cái thuộc tính css của html! rem thì phụ thuộc vào html, còn em sẽ phụ thuộc vào cái thuộc tính css gần nó nhất mà chưa font-size. Mặc định font-size: 100% sẽ bằng 16px Sau này vd như muốn sửa font-size của cả trang web và nếu đặt đơn vị là em, thì phải đi sửa ở rất nhiều nơi, tại vì em nó phụ thuộc vào thẻ font-size gần nó nhất. Vì vậy chúng ta mới sử dụng rem để nó phụ thuộc vào một cái duy nhất là html

vw, vh: v là viewport, w là width, h là height

14. CSS PADDING Padding là lớp đệm, để tránh nhầm lẫn với margin, thì padding nó đệm thêm về 4 hướng theo hướng mà mình muốn : top bottom, right, left| padding-top, padding-left ... Nhưng khi mà mình muốn tăng lên nhanh và tránh dài dòng thì chỉ sử dụng mỗi padding: 10px vd vậy.

Còn nếu trong paading có 2 số: padding 10px 12px có nghĩa là top & bottom là 10px, left & right là 12px.

padding: 10px 12px 8px có nghĩa là top 10, left & right 12, bottom 8

padding: 10px 12px 8px 6px có nghĩa là top 10, right 12, bottom 8, left 6 theo chiều kim đồng hồ.

15. CSS BORDER Là đường viền ôm element, mình có thể set border gồm có border-width, border-style, border-color Khi mình sử dụng padding và border thì kích thước của thực thể cũng sẽ tăng theo. Viết tắt của border: border: 10px solid red; nếu ko có màu thì bth, ko có kích thước thì mặc định của solid là 2 px,

16. CSS MARGIN Tạo một khoảng trắng giữa các thực thể như việc ta cầm thực thể và kéo nó ra. Việc sử dụng tương tự như padding. sở thích về tìm hiểu.

17. BOX-SIZING   
Khi một element có kích thước yêu cầu là 100, nhưng khi set content nó quá sát so với lề, ta thực hiện thêm padding để đẩy content ra, và sau đó chỉnh sửa lại kích thước của element, sau đó thêm border rồi lại quay lại chỉnh sửa kích thước của content, làm như vậy sẽ rất tốn công, thay vào đó mình có một thuộc tính đó là Box-sizing: có các thuộc tính sau: content-box: thuộc tính như cũ, như ban đầu; border-box: fit sao cho nó vừa đủ với kích thước mình yêu cầu; unset: hủy các thuộc tính, hoặc là ghi đè.

18. CSS Background-clip Nó sẽ quyết định là background sẽ được đổ từ ranh giới nào vào trong: vd background-clip: content-box; padding-box, border

19. Css background-image   
background-image: url() // copy link vào background-size: width, height; background-repeat: no-repeat; 2 ảnh được luôn, nhưng mỗi ảnh cách nhau bởi dấu phẩy: url () , url(); Ảnh nào được viết trước nó sẽ hiển thị lên trên.  
  
Không sử dụng hình ảnh mà sử dụng dãy màu chuyển: background-image: linear-gradient(0 (hướng màu), màu bắt đầu, màu kết thúc); Khi giá trị góc của nó là 0 thì chỉ cần viết 0, khi có độ lớn thì phải thêm độ: 180deg, 90deg.   
  
Sử dụng màu trong suốt: rgba() để thực hiện việc dùng 2 ảnh song song, ảnh là link trong url và ảnh là linear-gradient| Sử dụng dev tool để lấy giá trị màu của trang web mà mình làm theo hoặc là lấy màu của nhà cung cấp. Nếu muốn lấy mã màu thì bấm lên sau khi tìm màu, giá trị màu từ 0 tới 255 và cái a cuối cùng là độ trong suốt, mình muốn lấy độ trong suốt thì cho giá trị của nó từ 0 tới bé hơn 1

20. Css background-size keyword: gồm có contain và cover,   
contain thì nó lấy chiều dài nhất trong dài hoặc ngang theo size của trang web mà mình nhưng đảm bảo ko che khuất và cắt xén ảnh.   
cover lấy cạnh dài hơn bứa ảnh là lấp 100%, chấp nhận che khuất hình ảnh. Nhưng không hở ra khoảng trắng, còn contain thì có khoảng trắng.

21. Css background-origin sử dụng cho đổ hình ảnh, có thể đổ từ border, hoặc là đổ từ padding tùy mình tùy chỉnh:

22. Css background-position   
Sử dụng để điều chỉnh vị trí của ảnh: vd như background-position: top right; Khi viết một giá trị thì nó sẽ tự hiểu là center, nếu cho một giá trị xác định thì khoảng cách sẽ là từ mép trái đến bức ảnh, còn trên và dưới sẽ tự căn giữa. Khi mình muốn đặt ở góc nhưng cách ra so với mép một khoảng cách thì thêm vào sau đó một giá trị cố định : top 20px right 20px thì nó sẽ cách mép trên và bên phải 20px, sử dụng dấu âm (-) thì nó sẽ chạy ngược lên so với chiều mình muốn. Trường hợp sử dụng để cách ra theo giá trị xác định vd background-position: 50px 100px; thì nó sẽ hiểu là góc trái trên cùng là giao điểm của trục xy, 50 sẽ cách trục y là 50px, còn 100 thì sẽ cách trục x 100px. Muốn di chuyển ngược ra thì sử dụng dấu âm.

23, CSS background shorthand background: color, url () no-repeat center / contain; bấm vào background nó sẽ hiện lên cái thuộc tính giải thích của background cho mình và dưới cùng có cái MDN reference bám vào và open nó sẽ có demo và syntax, hướng dẫn sd đồ này kia, nên vào đọc cho kỹ.

24. Css functions: var (sử dụng để dùng biến); linear-gradient (sử dụng làm dải màu); rgba (sử dụng để tạo màu có độ trong suốt); rgb (sử dụng màu đặc)

Calc () tính toán được, calc (int px) nó sẽ tạo ra cái hình vuông, hoặc có thể dùng cả tương dối và tuyệt đối như 80% + 20px attr () cái này là attribute có thể lấy được giá trị của các attribute khác như href, ... <a href = "fullstack.edu.vn"> Học lập trình </a> sau đó lên style a:: after { content: "(" attr( href ) "); } thì khi mình cập nhật cái link trong href thì nó sẽ tự động cập nhật Tạo chữ trong một cái box .box::after { content: This is a box; }

25. Css pseudo class: lớp giả :root tham chiếu đến chính phần tử gốc của file html , cặp thẻ html :hover h1:hover { định nghĩa các thuộc tính css vào trong element nhưng chỉ khi nào mình di chuột vào thì css đó mới hiện lên} :active h1:active { css thuộc tính vào element nhưng khi nào mình ấn và giữ chuột vào trong đó thì thuộc tính đó mới hiện lên} :first-child or :last-child vd như ul có các thuộc tính con là li, thì first child sẽ chọn con đầu tiên và last sẽ chọn con cuối cùng. li:first-child { color: red;} thì thẻ li đầu tiên sẽ đổi màu Chủ động vào gg search css pseudo classes để tìm hiểm thêm về các lớp giả khác.

27. **Css Position: relative**: Viết như cú pháp trên nhưng đặt sau là một dấu chấm phẩy, sau đó top: 100px thì nó sẽ dịch xuống và đè lên cái thuộc tính khác luôn (khác với margin là nó sẽ di chuyển tất cả các thuộc tính xuống và không ghi đè lên các thuộc tính khác. Top left... sẽ không hoạt động nêu ko có position. Relative là vị trí tương đối và lấy chính mình làm gốc tọa độ và không phụ thuộc vào 1 đối tượng nào khác ở bên ngoài cả.  
  
 h1 {position:relative; top 10px; } thì nó sẽ ghi đè xuống dưới 10px

28. **Css Position: absolute** Trường hợp sử dụng 2 đối tượng, một đối tượng con di chuyển xung quanh một đối tượng cha, thì khi cần làm như vậy mình sẽ cho mọt đối tượng làm thẻ cha, có position là relative, thì thẻ con sẽ lấy thẻ cha làm gốc tọa độ do nó tìm thuộc tính position gần nhất để làm thẻ cha. Thẻ con không phân biệt thuộc tính của thẻ cha là gì, miễn là 1 trong 4 thuộc tính của position thôi. Không thể sử dụng ưu tiên cả top cả bottom, do nó chỉ ưu tiên top left thôi. vd khi mình bấm vào thông báo sẽ tạo ra 1 lớp phủ mờ che toàn bộ trang web, cho rằng lớp phủ đó là lớp con, sau đó bỏ width, height và cho top bottom right left bằng 0 thì nó sẽ phủ hết lớp cha. và nên sử dụng màu rgba() để nó có một lớp nhìn xuyên qua

29. **Css position: fixed** dùng để frezze header và footer Khi một đối tượng phụ thuộc vào một dối tượng khác thì position là absolute, còn khi một đối tượng phụ thuộc vào trang web thì position là fixed tạo một cái header, sau đó cho nó thuộc tính position: fixed rồi chỉnh margin cho h1 là có thể tạo frezee, còn khi mún header thành footer thì cho bottom nó bằng 0

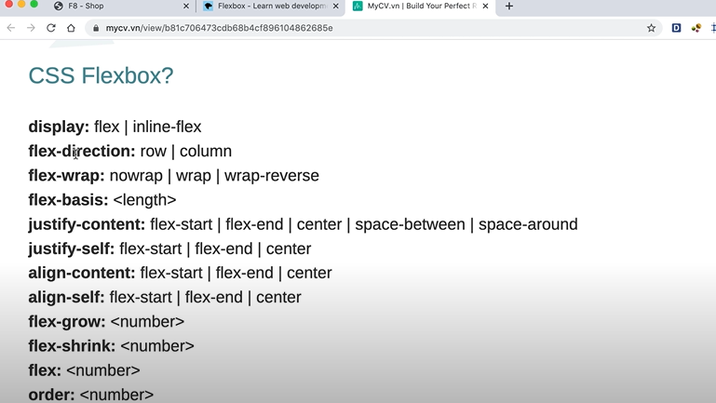
30. Css position: sticky sử dụng làm menu, header hoặc start bar bám dính,   
Bám dính là khi nó ở dưới, mình không thấy nó, cho tới khi mình lướt thấy nó và lướt lên thì nó sẽ dính vào ở trên mà không trôi đi. Không hỗ trợ trên nhiều trình duyệt. Có nghĩa là nó sẽ nằm cách top 1 khoảng n px, sau đó mình lướt xuống tới nó thì nó sẽ frezze sử dụng cho safari: position: -webkit-sticky;

31. Thuộc tính css trong flexbox display: có sử dụng flexbox hay không flex-direction: điều chỉnh phương hướng của main axis

khi sử dụng display: flex mặc định của flex direction là row có nghĩa là năm ngang, collum là dọc.

flex-wrap: xuống dòng, còn wrap-reverse là lên dòng (đảo cross start/end) flex-basis: <length> set kích thước cho main size

justify-content: căn item lên trên, giữa, xuống dưới theo main axis justify-sefl: khi mà set justify-content cho flex container thì tương tự như set các thuộc tính đó cho flex item và tương tự như việc set justify-sefl cho flex item.

[[](https://www.messenger.com/messenger_media/?thread_id=100016265675586&attachment_id=1108258133454499&message_id=mid.%24cAABa9tn8w0KJGzJ0NGC6e0rYLcOl)](https://www.messenger.com/messenger_media/?thread_id=100016265675586&attachment_id=1108258133454499&message_id=mid.%24cAABa9tn8w0KJGzJ0NGC6e0rYLcOl)

Tuy là học xong css flex box nhưng mà mình vẫn có những cái chưa hiểu nó dùng để làm cái gì, và muốn code ra một cái giao diện như vậy thì phải làm sao, trong đầu không có cách nào để tiến hành do ko thuộc bài thì làm sao mà làm được, bản thân việc làm code này thì mình phải thực hành nhiều để khi làm font end mình nhớ tất cả các thuộc tính để làm, cho nên phải học lại và code lại thui pà con ơi ơi.

35. CSS BEM là gì, đặt tên theo tiêu chuẩn BEM. #BEM là tiêu chuẩn đặt tên class khi viết CSS

## Ý nghĩa: - Viết tắt của Block Element Modifier - Block là khối - Element là thành phần trong khối - Modifier là bổ xung ý nghĩa cho "Block" hoặc "Element"

## Tại sao phải dùng BEM - Mỗi người một kiểu, css đè - Tạo hệ thống

## Cú pháp .block

.block\_\_element

.block--modifier

.block\_\_element--modifier

## Tính ứng dụng - Xây dựng layout website - Xây dựng thành phần website.

**36. VÍ dụ dùng BEM tạo buttons** Sử dụng .(tên class) để viết lệnh div class cho nó nhanh

Nút bấm không phải lúc nòa cũng dùng thẻ buttons, có thể dùng a hoặc div

border-radious: bo tròn nút trick: nút cạnh nhau, css thằng bên cạnh bằng cách lấy 2 class cộng nhau, rồi css cho nó thì các thuộc tính kế theo sẽ được css.

Nút có 2 trạng thái Enable và Disable, bấm được và không bấm được.

Nút bấm được có một con trỏ khi mình di vào. cursor: pointer;

**39. Dựng source base**

Bao gồm file main.css, base.css, html, img, font.

Tạo html cơ bản chuẩn bị code

Vào inspect xem có link được file main.css & base.css không.bõ

**40. Reset Css:**

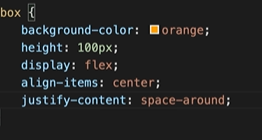
Không sử dụng thuộc tính css mặc định của trình duyệt, vào normalizer để lấy link xóa css.

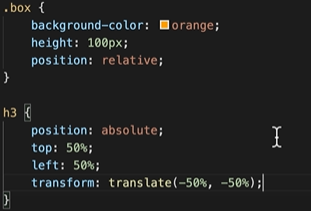
**41. Dựng Css base.**

* **Căn giữa trong CSS**Ví dụ khi có một element nằm trong một box, thì khi mình set text-align: center thì nó mới chỉ chạy ra giữa của top thôi, chưa có gì thay đổi nhiều còn khi mình muốn cho nó vào tâm luôn thì thêm một cái là **line-height** bằng với chiều cao của element chứa nó.



Khi sử dụng display flex cho cha và margin auto cho con, nó sẽ tự động căn giữa, có nhiều con thì căn giữa và căn đều.   
Hoặc có thể làm như vầy:

  
Khi làm không được thì sử dụng position



* **Hiển thị ảnh dự phòng khi ảnh chính bị lỗi**

Trong img thêm một cái attribute là onerror=”this.src=’ gán link vào đây, nên là link local’ ”

Nếu là ảnh của nền thì đặt sau link đầu là một link ảnh nữa, nếu ảnh đầu bị lỗi nó sẽ lấy ảnh thứ hai.

* **Xây dựng web The brand**

1. **Phân tích web**
2. **Dựng base (xây móng)**
3. **Xây dựng từng phần theo phân tích**

Việc đầu tiên là phân tích chứ không phải là code, nếu mà phân tích thì phải biết gọi tên các thành phần trong web đó là cái gì. Thì suy ra trong cuộc sống cũng vậy, khi mình phân tích bất cứ thứ gì trong bất cứ lĩnh vực gì, mình phải biết được nó là cái gì (ít nhất là về mặt tên gọi) nó có những thuộc tính gì, nhận diện nó như thế nào? Khi nào mình nhìn vào cái những thuộc tính của vật này thì mình biết nó là vật A, khi nào mình nhìn vào thuộc tính kia thì mình biết nó là vật B.

Những thành phần thường gặp trong giao diện web site:

1. Header
2. Navigation
3. Breadcrumb: phần địa chỉ từ trang lớn vào trang nhỏ. Hướng cho mình biết là mình đang ở đâu. 
4. Sidebar: một thanh đặt ở một phía bên màn hình.
5. Bannner quảng cáo.
6. Slider: hình ảnh
7. Content: nội dung.
8. Footer.

**PHÂN TÍCH VÀ XÂY DỰNG WEB THE BRAND**

1. Phân tích: trang sử dụng để làm cái gì => One page – thường sử dụng làm trang giới thiệu.
   1. Header
   2. Slider: show những gì nổi bật
   3. Content Giới thiệu - slogan, about (content nổi bật)
      1. About
      2. Tour
      3. Contact
      4. Image
      5. footer
   4. Footer

Làm subnav: nguyên tắc làm là cho nó hiện ra cái đã, làm đẹp đầy đủ rồi sau đó ẩn nó và cho nó hover sau.

RESPONSIVE WEBSITE

