TRƯỜNG CAO ĐẲNG LƯƠNG THỰC THỰC PHẨM

THIẾT KẾ WEB BẰNG HTML & CSS

Ths. Phạm Văn Hùng

ĐÀ NẴNG 9 -2020

BÀI 1: TỔNG QUAN VỀ HTML VÀ CSS

Thời gian:20 giờ
Mục tiêu :

- Hiểu được lịch sử phát triển của ngôn ngữ HTML
- Biết được những ứng dụng thực tế của ngôn ngữ HTML
- Nắm được các bước để thiết kế một trang Web

Nội dung:

1. Giới thiệu về quá trình phát triển Web

1.1. Làm việc với Web ứng dụng

Việc phát triển các ứng dụng trên nền Web hiện nay là 1 xu hướng được ưu tiên hàng đầu, việc thiết kế 1 trang web có thể chia thành 2 phần: Frontend và Backend.

Phần Frontend (phần giao diện của trang web)

Phần Backend (phần giao tiếp của trang web với cơ sở dữ liệu)

Các ngôn ngữ dùng để lập trình web hiện nay phổ thông là: JavaScript, Java, Python, PHP, C#, Ruby, HTML, SQL, MySQL, CSS.

Để làm phần Frontend phổ biến nhất vẫn sử dụng HTML và CSS, trong mô đun này chúng ta sẽ học 2 ngôn ngữ này để làm phần Frontend

1.2. Giới thiệu về HTML và CSS

a) HTML

HTML (Hyper Text Markup Language) tạm dịch là ngôn ngữ đánh dấu siêu văn bản. Người ta thường sử dụng HTML trong việc phân chia các đoạn văn, heading, links, blockquotes,…

HTML được tạo ra bởi Tim Berners-Lee, một nhà vật lý học của trung tâm nghiên cứu CERN ở Thụy Sĩ. Hiện nay, HTML đã trở thành một chuẩn Internet được tổ chức W3C (World Wide Web Consortium) vận hành và phát triển. Bạn có thể tự tìm kiếm tình trạng mới nhất của HTML tại bất kỳ thời điểm nào trên Website của W3C.

Phiên bản đầu tiên của HTML xuất hiện năm 1991, gồm 18 tag HTML. Phiên bản HTML 4.01 được xuất bản năm 1999. Sau đó, các nhà phát triển đã thay thế HTML bằng XHTML vào năm 2000.

Đến năm 2014, HTML được nâng cấp lên chuẩn HTML5 với nhiều tag được thêm vào markup, mục đích là để xác định rõ nội dung thuộc loại là gì (ví dụ như: <article>, <header>, <footer>,…).

Ưu điểm của HTML

HTML được sử dụng để tạo bố cục, cấu trúc trang web. Nó có một số ưu điểm sau:

- Có nhiều tài nguyên hỗ trợ với cộng đồng người dùng vô cùng lớn
- Có thể hoạt động mượt mà trên hầu hết mọi trình duyệt hiện nay
- Học HTML khá đơn giản
- Các markup sử dụng trong HTML thường ngắn gọn, có độ đồng nhất cao
- Sử dụng mã nguồn mở, hoàn toàn miễn phí
- HTML là chuẩn web được vận hành bởi W3C
- Dễ dàng để tích hợp với các loại ngôn ngữ backend (ví dụ như: PHP, Node.js,…)

Nhược điểm

Bên cạnh ưu điểm, HTML cũng có các nhược điểm nhất định. Cụ thể như sau:

- Chỉ được áp dụng chủ yếu cho web tĩnh. Nếu muốn tạo các tính năng động, lập trình viên phải dùng thêm JavaScript hoặc ngôn ngữ backend của bên thứ 3 (ví dụ như: PHP)

- Mỗi trang HTML cần được tạo riêng biệt, ngay có khi có nhiều yếu tố trùng lặp như header, footer.
- Khó để kiểm soát cách đọc và hiển thị file HTML của trình duyệt (ví dụ, một số trình duyệt cũ không render được tag mới. Do đó, dù trong HTML document có sử dụng các tag này thì trình duyệt cũng không đọc được).
- Một vài trình duyệt còn chậm cập nhật để hỗ trợ tính năng mới của HTML

b) CSS

CSS (Cascading Style Sheets) đó là một ngôn ngữ định kiểu. Điều này có nghĩa là nó cho phép chúng ta áp dụng kiểu có chọn lọc cho các phần tử trong tài liệu HTML, nói 1 cách khác CSS giúp chúng ta định dạng các phần tử trên trang HTML.

CSS là ngôn ngữ tạo phong cách cho trang web – Cascading Style Sheet language. Nó dùng để tạo phong cách và định kiểu cho những yếu tố được viết dưới dạng ngôn ngữ đánh dấu, như là HTML. Nó có thể điều khiển định dạng của nhiều trang web cùng lúc để tiết kiệm công sức cho người viết web. Nó phân biệt cách hiển thị của trang web với nội dung chính của trang bằng cách điều khiển bố cục, màu sắc, và font chữ.

CSS được phát triển bởi W3C (World Wide Web Consortium) vào năm 1996, vì một lý do đơn giản. HTML không được thiết kế để gắn tag để giúp định dạng trang web. Bạn chỉ có thể dùng nó để "đánh dấu" lên site.

Những tag như <font> được ra mắt trong HTML phiên bản 3.2, nó gây rất nhiều rắc rối cho lập trình viên. Vì website có nhiều font khác nhau, màu nền và phong cách khác nhau. Để viết lại code cho trang web là cả một quá trình dài, cực nhọc. Vì vậy, CSS được tạo bởi W3C là để giải quyết vấn đề này.

Mối tương quan giữa HTML và CSS rất mật thiết. HTML là ngôn ngữ markup (nền tảng của site) và CSS định hình phong cách (tất cả những gì tạo nên giao diện website), chúng là không thể tách rời.

CSS về lý thuyết không có cũng được, nhưng khi đó website sẽ không chỉ là một trang chứa văn bản mà không có gì khác.

1.3. Các công cụ để phát triển Web

Hiện nay để phát triển 1 trang Web chúng ta cần khá nhiều phần mềm, quan trọng nhất phải kể đến các trình soạn thảo và biên dịch Web như:

- Sublime Text
- Visual Studio Code
- PHP designer
- Dreamweaver
- NotePad ++

Nhưng Sublime Text là 1 lựa chọn tối ưu cho lập trình viên bởi vì nó hỗ trợ rất nhiều Plugin giúp chúng ta tăng tốc độ soạn code và bắt lỗi rất tốt.

Kế đến là các phần mềm hỗ trợ việc lấy màu, đo khung ảnh như: FSCapture, Pixie, PhotoShop

2. Cách viết mã, kiểm tra và triển khai một trang Web

2.1. Cú pháp của HTML

Một tập tin HTML sẽ chứa các phần tử HTML và được lưu lại với đuôi mở rộng là .html hoặc .htm. Các tập tin HTML sẽ được đọc bởi các trình duyệt web (Browser) như Firefox, Chrome, Cốc Cốc,…

Mã nguồn HTML của một trang web tối thiểu phải trông như sau:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Tiêu đề</title>
```

```
  </head>
  <body>
    <h1>Nội dung chính ở đây</h1>
  </body>
</html>
```

Dòng đầu tiên <!DOCTYPE html> báo hiệu cho trình duyệt biết rằng mã nguồn này sử dụng chuẩn HTML (HTML standard) thuộc phiên bản nào, Còn chúng ta từ giờ trở đi sẽ dùng DocType này để trình duyệt tự động áp dụng chuẩn mới nhất (hiện nay là HTML5) nhằm hiển thị đúng những thẻ và thuộc tính mới, thậm chí ảnh hưởng đến cả JavaScript:

Phần <head> dùng để chứa các thẻ lệnh cài đặt cho trang web, nó không được hiển thị lên màn hình. Phần văn bản bên trong cặp thẻ <title> sẽ được hiển thị trên tab hoặc trên thanh tiêu đề của cửa sổ trình duyệt.

Nội dung chính mà chúng ta sẽ dành hầu hết thời gian để làm việc nằm trong phần <body>. Trước hết các bạn cần nhớ những đặc điểm sau đây của HTML:

"Lệnh" của HTML gọi là "thẻ" (tag)

Thẻ HTML có dạng < rồi đến tên-thẻ sau đó là >, ví dụ <tên-thẻ>, , <img>…

Đa số các thẻ yêu cầu phải có thẻ mở và thẻ đóng để bao bọc nội dung bên trong, ví dụ: Bold, Italic. Thẻ đóng có cùng tên với thẻ mở cộng thêm dấu xuyệt / đằng trước.

Một số thẻ có thể đứng một mình là đủ, như thẻ <img>, <br>, <hr>, gọi là "thẻ đơn"

Các thẻ có thể lồng vào nhau (nested tag)

Các loại cặp thẻ có thể chứa nội dung là những thẻ khác để tạo nên một hiệu ứng kết hợp, ví dụ Bold and Italic.

Lưu ý rằng các cặp thẻ mở và thẻ đóng phải tương ứng với nhau, thẻ mở bọc ở ngoài thì thẻ đóng cũng phải bọc ở ngoài. Đối với một số trường hợp đơn giản như Bold & Italic thì trình duyệt có thể cố gắng hiển thị đúng mặc dù thứ tự thẻ đóng là sai. Tuy vậy nếu gặp những layout phức tạp hơn thì trình duyệt cũng bó tay, hậu quả sẽ là một trang web bị "bể layout".

Không phân biệt chữ hoa và chữ thường (case insensitive)

Tên các thẻ có thể viết thường hoặc in hoa đều có tác dụng như nhau

Cài đặt hành vi cho thẻ thông qua "thuộc tính" (attribute)

Mỗi thẻ HTML có một số thuộc tính (attribute) dùng để thay đổi hình dáng và hành vi của nó. Attribute có cú pháp tên="giá trị", bao gồm cả dấu nháy kép, thật ra dùng dấu nháy đơn ' cũng chạy được, nhưng các trainer ở CodeSchool sẽ yêu cầu các bạn dùng dấu nháy kép khi nộp bài tập để phù hợp với quy chuẩn của đa số công ty.

Ví dụ:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Thuộc tính của thẻ HTML</title>
  </head>
  <body>
    Văn bản: <input type="text" size="30" value="CodeSchool VN">
    <br><br>
    Mật khẩu: <input type="password" size="20" value="Secret">
    <br><br>
    Ngày tháng: <input type="date">
```

```
    <br><br>
    Sở thích:
    <input type="checkbox"> Đọc sách
    <input type="checkbox" checked> Ngồi thiền
    <input type="checkbox"> Chơi nhạc cụ
    <br><br>
    Giới tính:
    <input type="radio" name="gender"> Nam
    <input type="radio" name="gender"> Nữ
    <input type="radio" name="gender"> Khác
  </body>
</html>
```

Bỏ qua dấu xuống dòng và chỉ hiển thị một (01) ký tự khoảng trắng

Dấu ngắt dòng trong mã nguồn HTML sẽ được hiển thị thành một khoảng trắng, nếu muốn ngắt dòng thì phải dùng thẻ đơn
 (break), còn để ngắt thành đoạn văn mới thì phải dùng cặp thẻ Nội dung .

Tất cả dấu ngắt dòng và dấu khoảng trắng liên tiếp nhau sẽ được hiển thị thành một (01) dấu khoảng trắng. Nếu muốn chủ động hiển thị nhiều khoảng trắng thì phải dùng tới một loại ký tự đặc biệt (HTML Entity) gọi là "non-breaking space" (khoảng trắng không ngắt) viết là . HTML Entity là cú pháp dùng để hiển thị những ký tự không có trên bàn phím,   là một ký tự đặc biệt như thế, trình duyệt coi nó là một ký tự bình thường (nhưng vô hình), cho nên khi nó đứng kế tiếp nhau hàng loạt thì trình duyệt sẽ coi đây là một từ (word), và sẽ không ngắt dòng.

2.2. Cú pháp của CSS

Để tìm hiểu cú pháp CSS chúng ta hãy thử xem một ví dụ sau.

Ví dụ: Để định màu nền cho một trang web là xanh nhạt (light cyan) chúng ta dùng code sau:

+ Trong HTML: <body bgcolor="#00BFF3">

+ Trong CSS: body { background-color:#00BFF3; }

Nhìn qua ví dụ trên ít nhiều chúng ta cũng thấy được mối tương đồng giữa các thuộc tính trong HTML và CSS cho nên nếu bạn đã học qua HTML thì cũng sẽ rất dễ dàng tiếp thu CSS. Nhưng không sao cả, bây giờ hãy nhìn vào ví dụ của chúng ta và các bạn xem nó có giống với cấu trúc sau không nhé.

Cú pháp CSS cơ bản: **Selector { property:value; }**

Trong đó:

+ **Selector**: Các đối tượng mà chúng ta sẽ áp dụng các thuộc tính trình bày. Nó là các tag HTML, class hay id (chúng ta sẽ học về 2 thành phần này ở bài học sau). Ví dụ: body, h2, p, img, #title, #content, .username,…

Trong CSS ngoài viết tên selector theo tên tag, class, id. Chúng ta còn có thể viết tên selector theo phân cấp như để chỉ các ảnh ở trong #entry, chúng ta viết selector là #entry img, như vậy thì các thuộc tính chỉ định sẽ chỉ áp dụng riêng cho các ảnh nằm trong #entry.

Khi viết tên cho class, đôi khi sẽ có nhiều thành phần có cùng class đó, ví dụ như thẻ img và thẻ a cùng có class tên vistors nhưng đây lại là hai đối tượng khác nhau, 1 cái là ảnh của người thăm, 1 cái là liên kết tới trang người thăm. Nên nếu khi viết CSS ta ghi là .visitors { width:50 } thì sẽ ảnh hưởng tới cả hai thành phần.

Nên trong trường hợp này, nếu bạn có ý dùng CSS đó chỉ riêng phần ảnh thì chỉ nên ghi là img .visitors thôi.

Một lối viết tên selector nữa đó là dựa trên tên các thuộc tính có trong

HTML. Ví dụ trong HTML ta có đoạn mã như vầy: <input name="Search" type="Text" value="Key Word">. Để áp dụng thuộc tính CSS cho riêng ô tìm kiếm này chúng ta sẽ dùng selector input[name="Search"].

Ngoài việc viết tên selector cụ thể, chúng ta cũng có thể dùng một selector đại diện như * { color:red } sẽ tác động đến tất cả các thành phần có trên trang web làm cho chúng có text màu đỏ.

+ **Property**: Chính là các thuộc tính quy định cách trình bày. Ví dụ: backgroundcolor, font-family, color, padding, margin,…

Mỗi thuộc tính CSS phải được gán một giá trị. Nếu có nhiều hơn một thuộc tính cho một selector thì chúng ta phải dùng một dấu ; (chấm phẩy) để phân cách các thuộc tính. Tất cả các thuộc tính trong một selector sẽ được đặt trong một cặp ngoặc nhọn sau selector.

Ví dụ: body { background:#FFF; color:#FF0000; font-size:14pt }

Để dễ đọc hơn, bạn nên viết mỗi thuộc tính CSS ở một dòng. Tuy nhiên, nó sẽ làm tăng dung lượng lưu trữ CSS của bạn.

Ví dụ:
```
body {
        background:#FFF;
        bolor:#FF0000;
        font-size:14pt ;
      }
```

Đối với một trang web có nhiều thành phần có cùng một số thuộc tính, chúng ta có thể thực hiện gom gọn lại như sau:

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

+ **Value**: Giá trị của thuộc tính. Ví dụ: như ví dụ trên value chính là #FFF dùng để định màu trắng cho nền trang.

Đối với một giá trị có khoảng trắng, bạn nên đặt tất cả trong một dấu ngoặc kép. Ví dụ: font-family:"Times New Roman"

Đối với các giá trị là đơn vị đo, không nên đặt một khoảng cách giữa số đo với đơn vị của nó. Ví dụ: width:100 px. Nó sẽ làm CSS của bạn bị vô hiệu trên Mozilla/Firefox hay Netscape.

Chú thích trong CSS:

Cũng như nhiều ngôn ngữ web khác. Trong CSS, chúng ta cũng có thể viết chú thích cho các đoạn code để dễ dàng tìm, sửa chữa trong những lần cập nhật sau.

Chú thích trong CSS được viết như sau /* Nội dung chú thích */ Ví dụ:

```
/* Màu chữ cho trang web */ body {
        color:red
      }
```

Đơn vị CSS:

Trong CSS2 hỗ trợ các loại đơn vị là đơn vị đo chiều dài và đơn vị đo góc, thời gian, cường độ âm thanh và màu sắc. Tuy nhiên, sử dụng phổ biến nhất vẫn là đơn vị đo chiều dài và màu sắc. Sau đây là bảng liệt kê các đơn vị chiều dài và màu sắc dùng trong CSS

+ Chiều dài (Width):

Đơn vị	Mô tả	Đơn vị	Mô tả
%	Phần trăm	ex	1 ex bằng chiều cao của chữ x in thường của font hiện hành. Do đó, đơn vị này không những phụ thuộc trên kích cỡ font chữ mà còn phụ thuộc loại font chữ vì cùng 1 cỡ 14px nhưng chiều cao chữ x của font Times và font Tohama là khác nhau.
in	Inch (1 inch = 2.54 cm)		
cm	Centimeter		
mm	Millimeter		
em	1 em tương đương kích thước font hiện hành, nếu font hiện hành có kích cỡ 14px thì 1 em = 14 px. Đây là một đơn vị rất hữu ích trong việc hiển thị trang web.	pt	Point (1 pt = 1/72 inch)

2.3. Sử dụng Sublime Text để làm việc với file HTML và CSS

1. Download và cài đặt Sublime Text 3

Vào trang chủ của Sublime Text https://www.sublimetext.com/

- Nếu dùng hệ điều hành Windows thì click nút DOWNLOAD FOR WINDOWS để tải về và cài đặt

- Nếu dùng hệ điều hành khác thì click nút Download phía trên để tải về phiên bản phù hợp cho máy của mình

2. Cài thêm các plugin

Để dùng Sublime Text thuận tiện chúng ta nên cài thêm các plugin, trình tự thực hiện như sau:

Bước 1:

- Vào trang https://packagecontrol.io/

- Chờ 1 lát cho trang web load đầy đủ thông tin, trang web sẽ xuất hiện giống như hình dưới

- Click nút Install now sẽ xuất hiện

Bước 2:

- Khởi động Sublime Text

- Gõ **Ctrl + Shift + P** để mở cửa sổ lệnh của Sub, sau đó nhập lệnh `Install Package Control` ấn Enter để nó cài trình cài đặt các gói Plugin vào Sublime Text, sau đó muốn cài thêm Plugin cụ thể nào thì thực hiện bước 3.

Bước 3:

- Khởi động lại Sublime Text

- Gõ **Ctrl + Shift + P** để mở cửa sổ lệnh của Sublime Text, gõ chữ P sẽ xuất hiện

- Chọn dòng Package Control: Install Package sẽ xuất hiện cửa sổ mới
- Chọn 1 Plugin để cài chẳng hạn JavaScript & NodeJS Snippets

3. Các Plugin cần thiết cho lập trình viên Web

1. JavaScript & NodeJS Snippets
2. Emmet (gõ html:5 ấn tab sẽ tự động xổ ra)
3. Advanced New File
4. Bracket HighLighter
5. Color HighLighter
6. Color Picker
7. AutoPrefixer
8. HTML-CSS-JS Prettify
9. DocBlockr
10. CodeIntel
11. AutoFileName
12. View In Browser

Chi tiết có thể tham khảo công dụng của các plugin trên trang https://viblo.asia/p/sublime-text-se-tuyet-voi-hon-neu-dung-nhung-package-nay-Ljy5VdyVZra

Lưu ý:

1. Sau khi cài xong View In Browser bạn cần thiết lập thông số cho nó bằng cách; vào phần Preferences / Package Settings / View in Browser sau đó click vào lệnh Setting - default để mở tập tin này ra và copy toàn bộ nội dung của tập tin này và sau đó click chuột mở Setting - user và dán toàn bộ nội dung này vào sau đó đổi chữ firefox ở dòng cuối thành "browser": "chrome64"

4. Các phím tắt thường dùng của Sublime Text

1. Ctrl + X: Cắt dòng.
2. Ctrl + Shift + Enter: Thêm dòng phía trên con trỏ.
3. Ctrl + Enter: Thêm dòng phía dưới con trỏ.
4. Ctrl + Shift + ↑: Đưa dòng hiện tại lên trên 1 dòng.
5. Ctrl + Shift + ↓: Đưa dòng hiện tại xuống dưới 1 dòng.
6. Ctrl + L: Bôi đen cả dòng và đưa con trỏ xuống dòng tiếp theo.
7. Ctrl + D: Bôi đen từ đang được trỏ.
8. Ctrl + M: Đưa trỏ đến dấu đóng ngoặc gần nhất (ví dụ trong câu lệnh if-else).
9. Ctrl + Shift + M: Bôi đen toàn bộ nội dung trong cặp dấu ngoặc.
10. Ctrl + K: Xóa hết đến cuối dòng bắt đầu từ vị trí con trỏ.
11. Ctrl + K + Backspace: Xóa hết đến đầu dòng bắt đầu từ vị trí con trỏ.
12. Ctrl +]: Tab dòng hiện tại vào trong 1 tab.
13. Ctrl+ [: Lùi dòng hiện tại ra ngoài 1 tab.
14. **Ctrl + Shift + D: Nhân đôi dòng hiện tại hoặc khối lệnh được bôi đen.**

15. Ctrl + J: Nối dòng phía dưới xuống cuối dòng hiện tại của con trỏ.

16. Ctrl + /: Comment 1 dòng lệnh kiểu //.

17. Ctrl + Shift + /: Comment 1 khối dòng lệnh kiểu //.**

18. Ctrl + Y: Lấy lại những thao tác vừa bị Undo.

19. Ctrl + Shift + V: Dán và đưa con trỏ xuống cuối dòng.

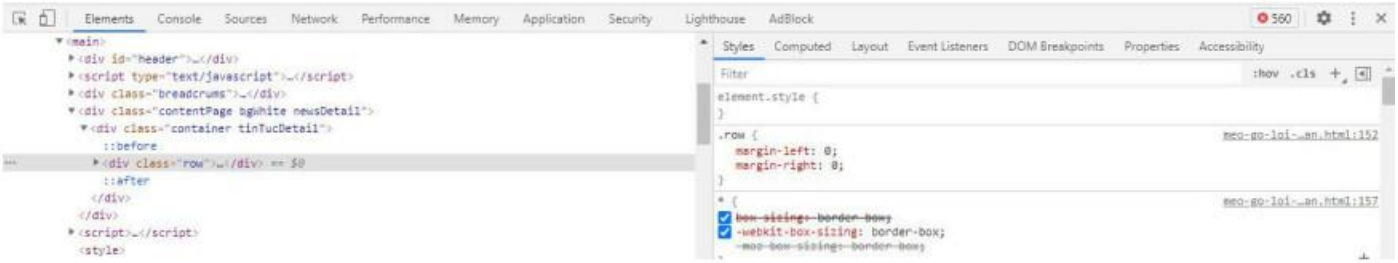
20. Ctrl + Space: Bật gợi ý.

21. Ctrl + U: Undo lặp lại những thao tác trước đó.

2.4. Cách kiểm tra, bắt lỗi và xác thực file HTML và CSS

Một phương pháp tốt nhất để kiểm tra, bắt lỗi và xác thực file HTML và CSS là dùng Chrome DevTools có sẵn trong trình duyệt Chrome

Để kiểm tra một phần tử chúng ta right click vào phần tử và chọn lệnh **Kiểm tra** hoặc bấm phím tắt Ctrl + Shift + I, sẽ xuất hiện giao diện như hình:



Trong đó thẻ Style sẽ chứa tất cả CSS có ảnh hưởng đến phần tử được chọn, chúng ta có thể thay đổi các giá trị các thuộc tính để thấy hiệu ứng trực tiếp lên Web

3. Sử dụng HTML để tạo cấu trúc trang Web

3.1. Cấu trúc của trang Web

Cấu trúc chuẩn của 1 trang web theo HTML5

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Bài thực hành 1</title>
    <link rel="stylesheet" href="BT1.css">
</head>
<body>

</body>
</html>
```

3.2. Cách viết mã phần đầu trang Web

Phần đầu của trang web nằm trong cặp thẻ <head> </head> nó có thể gồm nhiều thứ trong đó cơ bản chúng ta có các dòng:

<meta charset="UTF-8"> để có thể hiện đúng tiếng Việt trên trình duyệt

<meta name="viewport" content="width=device-width, initial-scale=1.0"> Dùng cho web đáp ứng (Responsive)

<title>Bài thực hành 1</title> Tiêu đề trang web

Và các thẻ Link

3.3. Cách viết mã cho liên kết

Ví dụ:

```
<script type="text/javascript" src="vendor/jquery.min.js"></script>
<script type="text/javascript" src="vendor/popper.min.js"></script>
<script type="text/javascript" src="vendor/bootstrap.js"></script>
<script type="text/javascript" src="1.js"></script>
<link rel="stylesheet" href="vendor/bootstrap.css">
<link rel="stylesheet" href="1.css">
```

Thông thường để link đến các file css, các thư viện Boostrap, hoặc các thư viện khác chúng ta dùng 2 loại thẻ <script) hoặc <link>

3.4. Cách viết mã phần nội dung trang Web

Phần nội dung do lập trình viên tự viết bằng các loại thẻ html sẽ được học phần kế tiếp, nằm bọc trong cặp thẻ <body>

4. Sử dụng CSS để định dạng các đối tượng của trang Web

4.1. Cách xác định cường độ của màu sắc trang Web

+ Màu sắc

Đơn vị	**Mô tả**
Color-name	Tên màu tiếng Anh. Ví dụ: black, white, red, green, blue, cyan, magenta,…
RGB (r,g,b)	Màu RGB với 3 giá trị R, G, B có trị từ 0 – 255 kết hợp với nhau tạo ra vô số màu.
RGB (%r,%g,%b)	Màu RGB với 3 giá trị R, G, B có trị từ 0 – 100% kết hợp.
Hexadecimal RGB	Mã màu RGB dạng hệ thập lục. Ví dụ: #FFFFFF: trắng, #000000: đen, #FF00FF: đỏ tươi.

4.2. Cách làm việc với CSS (Cascading Style Sheets)

Có 3 cách làm việc với CSS:

Cách 1: Style CSS Internal là style được tải lên mỗi khi trang web được refresh. Vì vậy nó tăng thời gian tải trang. Ngoài ra, bạn sẽ không dùng một style CSS cho nhiều trang vì nó chỉ áp dụng cho từng trang một. Tuy nhiên, lợi ích của style CSS Internal là khi mọi thứ đã đặt trong một trang thì nó dễ chia sẽ trang để xem trước hơn.

CSS Internal là cách dùng tất cả các mã CSS bên trong thẻ Style

Ví dụ làm trang web có màu nền trắng, đoạn văn bản chữ xanh lá, chúng ta sẽ thể hiện như sau:

```
<html>
<head>
  <title>Ví dụ</title>
        <style type=”text/css”> body { background-color:#FFF } p { color:#00FF00 }
        </style>
</head>
<body>
```

Lưu ý: Cách dùng này thẻ style phải được đặt trong phần head

Cách 2: Bên ngoài (liên kết với một file CSS bên ngoài) .Phương pháp dùng Style External là thuận tiên nhất. Mọi thứ được lưu trong file .css. Có nghĩa là bạn có thể tạo phong cách ở file khác áp dụng CSS vào trang bạn muốn. External style sẽ cải thiện thời gian tải trang rất nhiều.

Đây là cách làm được khuyến cáo, nó đặc biệt hữu ích cho việc đồng bộ hay bảo trì một website lớn sử dụng cùng một kiểu mẫu. Các ví dụ trong sách này cũng được trình bày theo kiểu này.

Cách 3: Style CSS Inline (Nội tuyến). Inline hoạt động với một yếu tố nhất định có tag <style>. Mỗi thành phần đều cần được tạo phong cách riêng, vì vậy đây không hẳn là cách tốt nhất và dễ nhất để xử lý CSS. Nhưng có thể khá tiện lợi, vì nếu bạn muốn thay đổi chỉ một yếu tố, nhanh chóng xem trước thay đổi, bạn không cần truy cập trực tiếp vào file CSS để chỉnh sửa mà sử dụng Inline CSS.

Đây là một phương pháp nguyên thủy nhất để nhúng CSS vào một tài liệu HTML bằng cách nhúng vào từng thẻ HTML muốn áp dụng. Và dĩ nhiên trong trường hợp này chúng ta sẽ không cần selector trong cú pháp.

Lưu ý: Nếu bạn muốn áp dụng nhiều thuộc tính cho nhiều thẻ HTML khác nhau thì không nên dùng cách này.

Ở ví dụ sau chúng ta sẽ tiến hành định nền màu trắng cho trang và màu chữ xanh lá cho đoạn văn bản như sau:

```
<html>
<head>
        <title>Ví dụ</title>
</head>
<body style="background-color=#FFF;">
<p style="color:green">^_^ Welcome To WallChúng ta's Blog ^_^</p>
</body>
</html>
```

5. Các thẻ Tiêu đề

Tiêu đề HTML là tiêu đề hoặc phụ đề mà bạn muốn hiển thị trên trang web, để tạo các tiêu đề chúng ta dùng các thẻ h1, h2, h3, h4, h5, h6

Ví dụ: đoạn code sau

```
<h1>Heading 1</h1>
<h2>Heading 2</h2>
<h3>Heading 3</h3>
<h4>Heading 4</h4>
<h5>Heading 5</h5>
<h6>Heading 6</h6>
```

sẽ cho kết quả

Heading 1

Heading 2

Heading 3

Heading 4

Heading 5

Heading 6

6. Các thẻ định dạng văn bản

a. Thẻ i hoặc em

Ví dụ: `<i>`This text is italic`</i>` sẽ cho kết quả *This text is italic*

Chúng ta cũng có thể thay thế thẻ i bằng thẻ em cũng có tác dụng định dạng chữ nghiêng

b. Thẻ b hoặc strong

cả 2 thẻ này đều có tác dụng định dạng chữ đậm

c. Thẻ u hoặc ins

cả 2 thẻ này đều có tác dụng định dạng chữ gạch chân

7. Các thẻ khối (block) và trên dòng (inline)

Mỗi phần tử HTML đều có giá trị hiển thị mặc định, tùy thuộc vào loại phần tử đó. Có hai cách hiển thị: dạng khối và dạng nội tuyến (trên 1 dòng)

a. Thẻ khối

Một phần tử cấp khối (A block-level element) luôn bắt đầu trên một dòng mới

Một phần tử cấp khối luôn chiếm toàn bộ chiều rộng có sẵn (trải dài sang trái và phải hết mức có thể)

Một phần tử cấp khối có một lề trên và một lề dưới, trong khi một phần tử nội tuyến thì không có các lề

Dưới đây là các phần tử cấp khối thường dùng trong HTML:

<p> <hr> <form> <nav> <section> <table>   <article> và các thẻ h, trong đó chúng ta thường dùng các thẻ:

+ Thẻ p: cho chúng ta xuất ra trình duyệt một đoạn văn bản

+ Thẻ div: Thẻ xác định một bộ phận hoặc một phần trong tài liệu HTML (một khối Web), Thẻ được sử dụng làm vùng chứa cho các phần tử HTML - sau đó được tạo kiểu bằng CSS hoặc được thao tác với JavaScript, Thẻ được tạo kiểu dễ dàng bằng cách sử dụng thuộc tính class hoặc id.

Ví dụ:

```
<!DOCTYPE html>
<html>
<head>
<style>
.myDiv {
  border: 5px outset red;
  background-color: lightblue;
  text-align: center;
}
</style>
</head>
<body>
        <h1>Phần tử div</h1>
        <div class="myDiv">
                <h2>Đây là  một heading nằm trong div</h2>
                <p>Đây là 1 đoạn văn bản nằm trong idv.</p>
        </div>
<p>Đây là đoạn văn bản nằm bên ngoài 1 div.</p>
```

</body>

</html>

Kết quả khi view trên trình duyệt

+ Thẻ hr: Đơn giản chỉ tạo 1 đường kẻ

b. Thẻ inline

Các phần tử (thẻ) dạng Inline thì không bắt đầu trên 1 dòng mới, Một phần tử inline chỉ chiếm nhiều chiều rộng khi cần thiết

Dưới đây là các phần tử Inline thường dùng trong HTML:

<textarea>, , <a>, <b>, <br>, <button>, , <i>, <img>, <label>, <select>

+ Thẻ span: Thẻ là một vùng chứa inline (nội tuyến) được sử dụng để đánh dấu một phần của văn bản hoặc một phần của tài liệu.

Thẻ dễ dàng được tạo kiểu bởi CSS hoặc được thao tác với JavaScript bằng cách sử dụng thuộc tính class hoặc id

Thẻ giống như phần tử , nhưng là phần tử block và là phần tử inline.

Ví dụ:

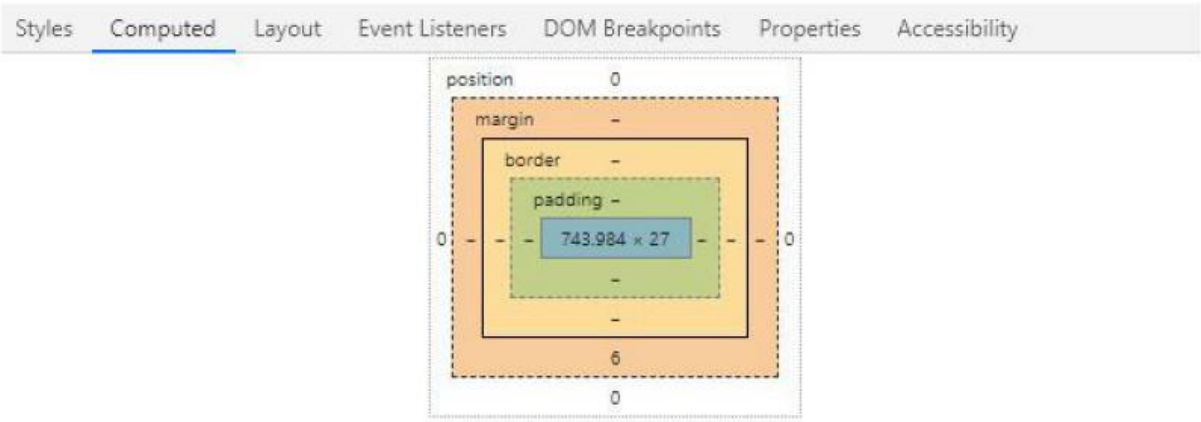
My mother has blue eyes and my father has dark green eyes.

+ Thẻ br:Đơn giản dùng để xuống dòng

8. Box Model

8.1. Giới thiệu về Box Model

Box Model là một kỹ thuật cơ bản nhất trong CSS Layout và được sử dụng để bạn mô tả về khoảng cách mà mỗi phần tử trên website được sở hữu, hay nói cách khác là kỹ thuật tinh chỉnh khoảng cách hiển thị cho mỗi phần tử trên website.



Kỹ thuật Box Model trong CSS bao gồm 4 phần quan trọng đó là:

- **Margin:** Khoảng cách tính từ bên ngoài của phần tử.

- **Border:** Đường viền của phần tử.
- **Padding:** Khoảng cách tính từ bên trong của phần tử.
- **Content**: Nội dung trong phần tử. (phần có màu xanh biển nhạt)

Để canh chỉnh 4 phần đó chúng ta sẽ dùng CSS, trong 4 thành phần trên thì phần content chúng ta sẽ không có thuộc tính CSS nào đại diện cả vì nó là nội dung trong phần tử. Còn 3 phần còn lại đều có các thuộc tính CSS để cài đặt.

8.2. Cách làm việc với kích cỡ và khoảng cách giữa các đối tượng

Để xử lý vấn đề này chúng ta dùng 2 thuộc tính Margin và Padding.

Padding:

Padding nghĩa là chúng ta sẽ thiết lập khoảng cách được tính từ phần Content trở ra viền của phần tử, đơn giản vậy thôi. Padding được khai báo trong CSS bởi thuộc tính padding với giá trị theo tuần tự top right bottom left (trên > phải > dưới > trái) và giá trị là số kèm theo đơn vị đo lường. (phần màu xanh lá cây).

CSS có các thuộc tính để chỉ định phần đệm (Padding) cho mỗi bên của phần tử:

- `padding-top`
- `padding-right`
- `padding-bottom`
- `padding-left`

Tất cả các thuộc tính padding có thể sử dụng các giá trị sau:

- **length** - chỉ định độ rộng padding tính bằng đơn vị px, pt, cm, …

- **%** - chỉ định padding theo tỉ lệ % độ rộng của phần tử mẹ chứa nội dung (content)

Ví dụ:

```
<!DOCTYPE html>
<html>
<head>
<style>
  div {
        border: 1px solid black;
        background-color: lightblue;
        padding-top: 50px;
        padding-right: 30px;
        padding-bottom: 50px;
        padding-left: 80px;
    }
</style>
</head>
<body>
        <h2>Cách dùng thuộc tính padding riêng biệt cho từng cạnh</h2>

        <div> div này có top padding là 50px, right padding 30px, bottom padding 50px, và left
  padding 80px.</div>
</body>
```

</html>

Khi View trên trình duyệt sẽ có kết quả

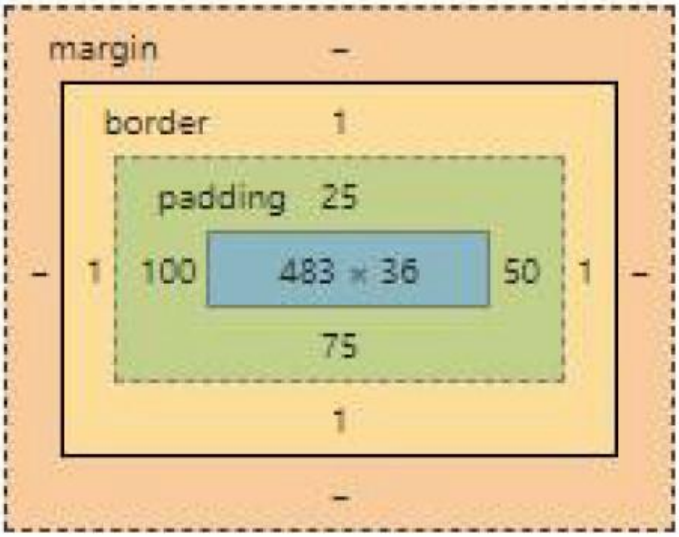
Cách dùng thuộc tính padding riêng biệt cho từng cạnh

div này có top padding là 50px, right padding 30px, bottom padding 50px, và left padding 80px.

Có 1 số tuỳ chọn cho padding như sau:

- Nếu thuộc tính padding có bốn giá trị, ví dụ:

padding: 25px 50px 75px 100px; (top 25px; right 50px; bottom 75px và left là 100px), khi kiểm tra phần tử chúng ta thấy.



- Nếu thuộc tính padding có ba giá trị, ví dụ:

padding: 25px 50px 75px; (top 25px, right và left là 50px và bottom 75px)

- Nếu thuộc tính padding có 2 giá trị, ví dụ:

padding: 25px 50px; (top và bottom là 25px; right và left là 50px)

- Nếu thuộc tính padding có 1 giá trị, ví dụ:

padding:27px; (cả 4 cạnh đều có padding 27px)

Margin

Trong khi Padding có nhiệm vụ tạo khoảng cách giữa phần Content với Border thì Margin sẽ có tác dụng tạo khoảng cách từ Border trở ra ngoài, nói dễ hiểu thì nó sẽ giúp bạn tinh chỉnh khoảng cách giữa các phần tử với nhau. Về cách dùng thì tương tự như Padding

Chúng ta xem ví dụ:

Code HTML

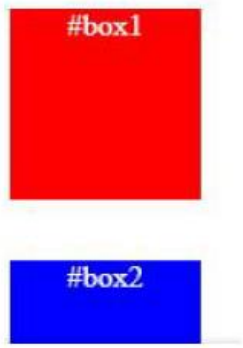
```
<div id="box1">
  #box1
</div>
<div id="box2">
  #box2
```

```
</div>
```

Code CSS

```
#box1 {
  background: red;
  color: #fff;
}
#box2 {
  margin: 32px 0 0 0; /* top right bottom left */
  background: blue;
  color: #fff;
}
```

Kết quả trên trình duyệt



Chúng ta thấy #box1 và #box2 có một khoảng trắng ? Đó là margin, hãy xem code CSS thì sẽ thấy ở #box2 có gắn thêm margin: 32px 0 0 0, tức là nó sẽ tạo ra một khoảng trắng ở trên đầu của #box2 và các mặt khác thì không tạo ra, cách viết tương tự như padding là 4 giá trị sẽ sắp xếp theo top right bottom left.

8.3. Cách thiết lập Borders và Backgrounds

Border

Border nghĩa là thuộc tính để bạn tạo viền cho phần tử và nó sẽ được khai báo bằng thuộc tính border trong CSS

Thuộc tính border này sẽ viết theo một số cấu trúc như sau:

1/ border: [size] [type] [color];

Ví dụ mình muốn tạo một cái viền, viền có kích thước 1px, kiểu trơn và màu viên là đỏ thì sẽ viết như sau: border: 1px solid red;

Trong border có hỗ trợ một số type như: solid (trơn) ,dotted (chấm chấm) , double (nét kép) ,groove (đường viền có rãnh 3D., ridge (đường viền 3D có gờ) , dashed(gạch gạch), inset (đường viền 3D lỏm vào) và outset (đường viền 3D lồi).

Giống như các thẻ trong Box Model khác, border cũng có các thẻ con là border-top, border-right, border-bottom và border-left

2/ border-style: [type]

Thuộc tính này chỉ loại đường viền khi hiển thị

3/ border-width: [size]

Thuộc tính này chỉ kích thước đường viền khi hiển thị

4/ border-color: [color]

Thuộc tính này chỉ màu đường viền khi hiển thị

5/ border-radius: [size]

Thuộc tính này làm cho 4 góc đường viền cong khi hiển thị

Background

Thuộc tính Background (nền) CSS được sử dụng để thêm hiệu ứng nền cho các phần tử.

Thuộc tính này được viết theo 1 số cách sau:

1/ background-color: [color]

Ví dụ: Làm cho nền trang web có màu xanh sáng

```
body {
  background-color: lightblue;
}
```

2/ Opacity

Thuộc tính opacity chỉ định độ mờ đục (opacity) / trong suốt (transparency) của một phần tử. Nó có thể nhận giá trị từ 0,0 - 1,0. Giá trị càng thấp, càng minh bạch:

Ví dụ:

```
<!DOCTYPE html>
<html>
<head>
<style>
div {
  background-color: green;
}

div.first {
  opacity: 0.1;
}

div.second {
  opacity: 0.3;
}

div.third {
  opacity: 0.6;
}
</style>
</head>
<body>

<div class="first">
  <h1>opacity 0.1</h1>
</div>

<div class="second">
```

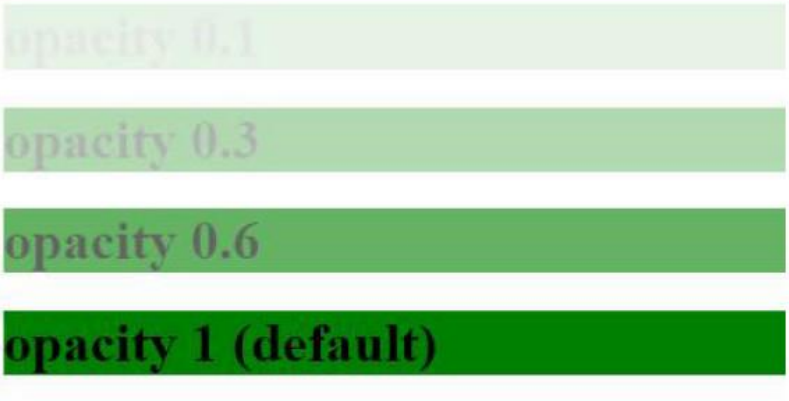
```
  <h1>opacity 0.3</h1>
</div>

<div class="third">
  <h1>opacity 0.6</h1>
</div>

<div>
  <h1>opacity 1 (default)</h1>
</div>

</body>
</html>
```

Khi View sẽ cho kết quả



3/ background-image: url(path)

Thuộc tính này dùng 1 ảnh để làm nền cho phần tử

Ví dụ: Lấy hình paper.gif làm nền

```
body {
  background-image: url("paper.gif");
}
```

9. Làm việc với danh sách và liên kết

9.1. Cách viết mã cho danh sách

Danh sách trong HTML có 2 dạng:

Danh sách có thứ tự

Để thể hiện danh sách có thứ tự ta sử dụng cặp thẻ: và , trong đó:

- là viết tắt của chữ "ordered list" có nghĩa là danh sách có thứ tự.
- viết tắt của chữ "list item" có nghĩa là mục của danh sách.

Ví dụ 1:

```
<ol>
   <li>Cá lóc kho tiêu</li>
   <li>Cá rô kho tộ</li>
```

```
  <li>Cá thu chiên xoài bằm</li>
  <li>Cá điêu hồng nấu ngót</li>
</ol>
```

Danh sách không có thứ tự

Để thể hiện danh sách không có thứ tự ta sử dụng cặp thẻ: và , trong đó:

- là viết tắt của chữ: unordered list có nghĩa là danh sách không có thứ tự

- viết tắt của chữ: list item có nghĩa là mục của danh sách.

Ví dụ 2:

```
<ul>
  <li>Trang chủ</li>
  <li>Giới thiệu</li>
  <li>Sản phẩm</li>
  <li>Dịch vụ</li>
  <li>Liên hệ</li>
</ul>
```

9.2. Cách định dạng cho danh sách

Đối với danh sách có thứ tự thì các phần tử trong danh sách ol tự động được đánh chỉ số : 1, 2, 3 ..., giống như trong Ví dụ 1 phía trên, khi xuất hiện trên trình duyệt sẽ là.

1. Cá lóc kho tiêu
2. Cá rô kho tộ
3. Cá thu chiên xoài bằm
4. Cá điêu hồng nấu ngót

Tuy nhiên chúng ta vẫn có thể định dạng lại bằng cách dùng thuộc tính **type** gán bằng a, A, i ... để thiết lập một số kiểu đánh số:

- a dùng chữ thường để đánh chỉ số
- A chữ in của bảng chữ cái (bắt đầu từ A) để đánh chỉ số
- i hoặc I đánh theo số la mã
- 1 kiểu mặc định (dùng số 1,2, ...)

Ngoài ra dùng thuộc tính **start** để thiết lập giá trị bắt đầu đánh số

Ví dụ:

```
<ol type="I" start="3">
  <li>Lúa </li>
  <li>Ngô </li>
  <li> Khoai</li>
</ol>
```

Đối với danh sách không có thứ tự các phần tử trong danh sách ul được đánh dấu đầu dòng bằng ký hiệu hình tròn

9.3. Cách viết mã cho liên kết

Các liên kết - link là một phần không thể thiếu cho mọi trang web. Bạn có thể thêm link dạng text(văn bản) hay dạng ảnh mà người dùng bấm chuột vào nó để chuyển hướng đến một trang web khác, một file khác.

Để tạo các liên kết chúng ta dùng thẻ a; Sử dụng thẻ a (anchor) để tạo link, liên kết trong văn bản HTML với các thuộc tính href để thiết lập URL chuyển đến và target thiết lập cách mở link.

Tạo liên kết (link) bằng thẻ <a> (Anchor - mỏ neo - tạo một điểm neo trên trang) trong sử dụng thuộc tính href (hyperlink reference) để chỉ ra địa chỉ đích mà link mở ra.

Ví dụ: để link tới trang elearning CD LTTP có địa chỉ http://hoctot.cfi-elearning.edu.vn/

```
<a href=" http://hoctot.cfi-elearning.edu.vn/">Hệ thống elearning CFI</a>
```

Thuộc tính target quy định vị trí mà link được mở ra. Giá trị là "_blank" sẽ mở link trong một tab mới của trình duyệt. Mặc định nó có giá trị "_self" mở link trong tab chứa trang gốc.

```
<a href=" http://hoctot.cfi-elearning.edu.vn/ " target="_blank">Hệ thống elearning CFI</a>
```

9.4. Cách tạo menu

Hầu hết các trang web ngày nay đều có phần menu và khi click vào menu sẽ trỏ ra một list các menu item ở phía dưới.

Menu đa cấp là gì?

Menu đa cấp là menu cung cấp danh sách các tùy chọn thường xuất hiện ở phần đầu tiên của mỗi trang web. Khi mục hiển thị được nhấp vào, các mục khác từ danh sách sẽ "thả xuống" list các danh sách con trong chế độ xem và người dùng có thể chọn từ các tùy chọn đó.

Tạo menu đa cấp cho phép người dùng lựa chọn một tùy chọn từ danh sách. Đây là một giải pháp không thể thiếu trong thiết kế trang web, đặc biệt là các website có số lượng chỉ mục nhiều, không thể sắp xếp toàn bộ trên giao diện

Các thành phần cần có của một menu đa cấp

Để tạo menu đa cấp nói riêng cũng như các loại menu khác nói chung thì các thành phần cũng tương tự gần như nhau.

+ Thẻ là một thẻ được dùng để gom nhóm nhiều phần tử. Thường được dùng để tạo bố cục cho website trong đó phải kể đến menu đa cấp. Thẻ gom nhóm các phần tử của menu vào một khu vực để tách với các phần khác.

+ Để tạo danh sách các menu item có trong menu đa cấp chúng ta cần phải sự dụng thẻ , thẻ này thường đi kèm vơi thẻ

Ngoài ra để có thể tạo một menu đa cấp hoàn chỉnh chúng ta cần phải sự dụng CSS để căn chỉnh các thành phần để nó có thể về đúng vị trí và trông đẹp mắt hơn.

Các bước để tạo 1 menu đa cấp

Bước 1: Tạo 1 folder (project) để chứa tất cả các file html và css

Bước 2: tạo file html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="menu021.css">
  <title>Menu 1</title>
</head>
<body>
  <div class="bar">
    <ul class="menungang">
      <li><a href="">Xã hội</a></li>
      <li><a href="">Thế giới</a></li>
      <li><a href="">Kinh tế</a></li>
```

```
            <li><a href="">Giáo dục</a></li>
            <li><a href="">Pháp Luật</a></li>
            <li><a href="">Công nghệ</a></li>
            <li><a href="">Khoa học</a></li>
            <li><a href="">Xe cộ</a></li>
        </ul>
    </div>
</body>
</html>
```

Bước 3: Menu ban đầu xuất hiện trên trình duyệt là một danh sách, để định dạng chúng ta phải dùng CSS (code menu021.css)

```
ul.menungang {
    display: flex;
    list-style: none; /* mất các bullet */
    width:900px;     /* Combo 3 dòng 4 5 6 để canh giữa các mục trên menu */
    margin-left: auto;
    margin-right: auto;
}

.bar {
    background-color:#2FA1B3;
    height: 40px;
    width: 100%;
}

ul.menungang a {
    color:white;
    text-decoration: none; /* bỏ gạch chân dưới các mục */
    font-family: segoe ui light; /* font cho chữ menu */
    text-transform: uppercase;   /* Đổi lên chữ hoa */
    font-size: 13px;
    padding: 10px 6px 12px 6px;   /* Theo chiều kim đồng hồ Top right bottom left */
    display: block;
   /* Nếu chỉ dùng padding-top thêm khoảng trống trên, mà ko thêm space vào thì cần display:block */
    transition: 0.4s; /* Thời gian khi di chuột qua chậm hơn nên mượt hơn */
}
ul.menungang a:hover {    /* Khi rê chuột qua một mục thì xử lý */
    background-color:#156672;
}
```

9.5. Một số thuộc tính CSS cần thiết khi xử lý menu

1. Display

- display: block; (element (phần tử) sẽ hiển thị như một khối, khi sử dụng giá trị block thành phần sẽ đứng một hàng độc lập so với thành phần trước và sau nó.)

- display: inline; (element sẽ hiển thị như một nội tuyến (inline, không ngắt dòng), đây là dạng mặc định.)

- display: inline-block; (element sẽ hiển thị như một khối, nhưng là một khối nội tuyến.)

- display: flex; (element sẽ hiển thị dưới dạng vùng chứa linh hoạt cấp khối, trãi thành dòng ngang)

2. list-style

Thuộc tính list-style thiết lập kiểu cho một danh sách. (mặc định bullet là dấu chấm tròn)

- list-style:square; (Xuất hiện bullet vuông)

- list-style: none; (Bỏ bullet)

3. text-decoration

- text-decoration: none; (bỏ gạch chân trên các link)

10. Định dạng CSS cho văn bản (Text)

10.1. Text color

Thuộc tính **color** (màu) được sử dụng để đặt màu của văn bản. ví dụ muốn định dạng màu xanh cho thẻ h1, chúng ta dùng css sau:

```
h1 {
  color: green;
}
```

Nếu chúng ta muốn định dạng màu nền của thẻ h1 là đen, và chữ màu trắng thì sử dụng

```
h1 {
  background-color: black;
  color: white;
}
```

10.2. Text Alignment

Thuộc tính **text-align** được sử dụng để đặt căn lề ngang của văn bản, Văn bản có thể được căn trái hoặc phải, căn giữa hoặc căn đều.

Ví dụ:

```
h1 {
 text-align: center;
}

h2 {
  text-align: left;
}

h3 {
  text-align: right;
}
```

10.3. Text Decoration

Thuộc tính **text-decoration** (trang trí văn bản) được sử dụng để đặt hoặc xóa trang trí khỏi văn bản.

Chúng ta sử dụng `text-decoration: none;` dùng để xóa gạch chân khỏi các liên kết (thẻ a)

10.4. Text Transformation

Thuộc tính **text-transform** được sử dụng để chỉ định chữ hoa và chữ thường trong văn bản.

Ví dụ:

```
p.uppercase {
  text-transform: uppercase;
}

p.lowercase {
  text-transform: lowercase;
}
```

11. Định dạng CSS cho font

11.1. Loại font

Để định dạng loại font sử dụng cho 1 phần tử chúng ta sử dụng thuộc tính **font-family**

Ví dụ: để định dạng thẻ p có văn bản bên trong sử dụng font Segoe UI

```
p {
  font-family: Segoe UI;
}
```

11.2. Kích thước font

Để định dạng kích font sử dụng cho 1 phần tử chúng ta sử dụng thuộc tính **font-size**

Ví dụ: để định dạng thẻ p có văn bản bên trong kích thước 14px

```
p {
  font-size:14px;
}
```

11.3. font-style

Để định dạng kiểu dáng font nghiêng sử dụng cho 1 phần tử chúng ta sử dụng thuộc tính **font-style**

Với cú pháp: `font-style: italic;`

11.4. font-weight

Để định dạng kiểu dáng font đậm sử dụng cho 1 phần tử chúng ta sử dụng thuộc tính **font-weight**

Với cú pháp: `font-weight: bold;`

12. Định dạng CSS cho chiều ngang & chiều cao phần tử

Thuộc tính chiều cao (high) và chiều rộng (width) CSS được sử dụng để đặt chiều cao và chiều rộng của một phần tử.

Các giá trị thường dùng:

Auto (mặc định)- Trình duyệt sẽ tự tính toán

Length - định nghĩa chiều cao/ bề ngang bằng các đơn vị như: px;

% - Xác định tỉ lệ phần trăm theo phần tử mẹ

13. Class và ID

Trong ngôn ngữ HTML, thuộc tính id & thuộc tính class được sử dụng để đặt tên (phân loại) các phần tử, mục đích là để tiện cho việc quản lý, định dạng các phần tử sau này.

13.1. ID

- Thuộc tính id dùng để đặt tên cho phần tử, tên này phải là duy nhất, không có trường hợp tên id của các phần tử bị trùng nhau (nếu so sánh một phần tử trong trang web giống như một người công dân Việt Nam thì tên id cũng giống như số CMND, nó dùng để xác định danh tính của phần tử).

- Để khai báo id cho một phần tử thì chúng ta đặt thuộc tính id vào bên trong thẻ mở của phần tử đó với cú pháp **id="tên id".** Sau khi đã khai báo, nếu muốn định dạng cho phần tử thì chúng ta chỉ cần gọi thẳng tên id của nó với cú pháp **#tên id**

Ví dụ:

```
<!DOCTYPE html>
<html>
<head>
   <title>Xem ví dụ</title>
   <meta charset="utf-8">
   <style type="text/css">
      #step{color:red;}
   </style>
</head>
<body>
   <p>Chức năng của thuộc tính id</p>
   <p id="step">Cách sử dụng thuộc tính id</p>
   <p>Tầm quan trọng của thuộc tính id</p>
</body>
</html>
```

13.2. Class

- Chức năng của thuộc tính class cũng gần giống với thuộc tính id, đó chính là dùng để đặt tên cho phần tử. Tuy nhiên, việc đặt tên class khác với tên id ở chỗ là với cùng một tên class thì chúng ta có thể dùng để đặt cho nhiều phần tử khác nhau (nếu so sánh một phần tử trong trang web giống như một người công dân Việt Nam thì tên class cũng giống như một cái biệt danh, mà biệt danh thì có thể dùng để đặt cho nhiều người khác nhau).

- Để khai báo class cho một phần tử thì chúng ta đặt thuộc tính class vào bên trong thẻ mở của phần tử với cú pháp class="tên class". Sau khi đã khai báo class, nếu muốn định dạng cho phần tử thì chúng ta chỉ cần gọi thẳng tên class của nó với cú pháp .tên class

Ví dụ:

```
<!DOCTYPE html>
<html>
<head>
  <title>Xem ví dụ</title>
  <meta charset="utf-8">
  <style type="text/css">
          .hello{color:red;}
  </style>
</head>
```

```
<body>
  <p>Chức năng của thuộc tính class</p>
  <p class="hello">Tầm quan trọng của thuộc tính class</p>
  <p>Cách sử dụng thuộc tính class</p>
  <p class="hello">Tìm hiểu cách đặt tên cho class</p>
  <p class="hello">Thuộc tính class là gì</p>
</body>
</html>
```

Lưu ý:

1. Tên ID hoặc Class không được bắt đầu bằng ký tự số, dùng chữ không dấu và viết liền nhau (không có khoảng trắng).

2. Một người có thể có nhiều cái biệt danh thì phần tử cũng tương tự như vậy, một phần tử có thể có nhiều tên class (nhớ thêm một dấu "khoảng trắng" nằm ngăn cách giữa các tên class)

Ví dụ:

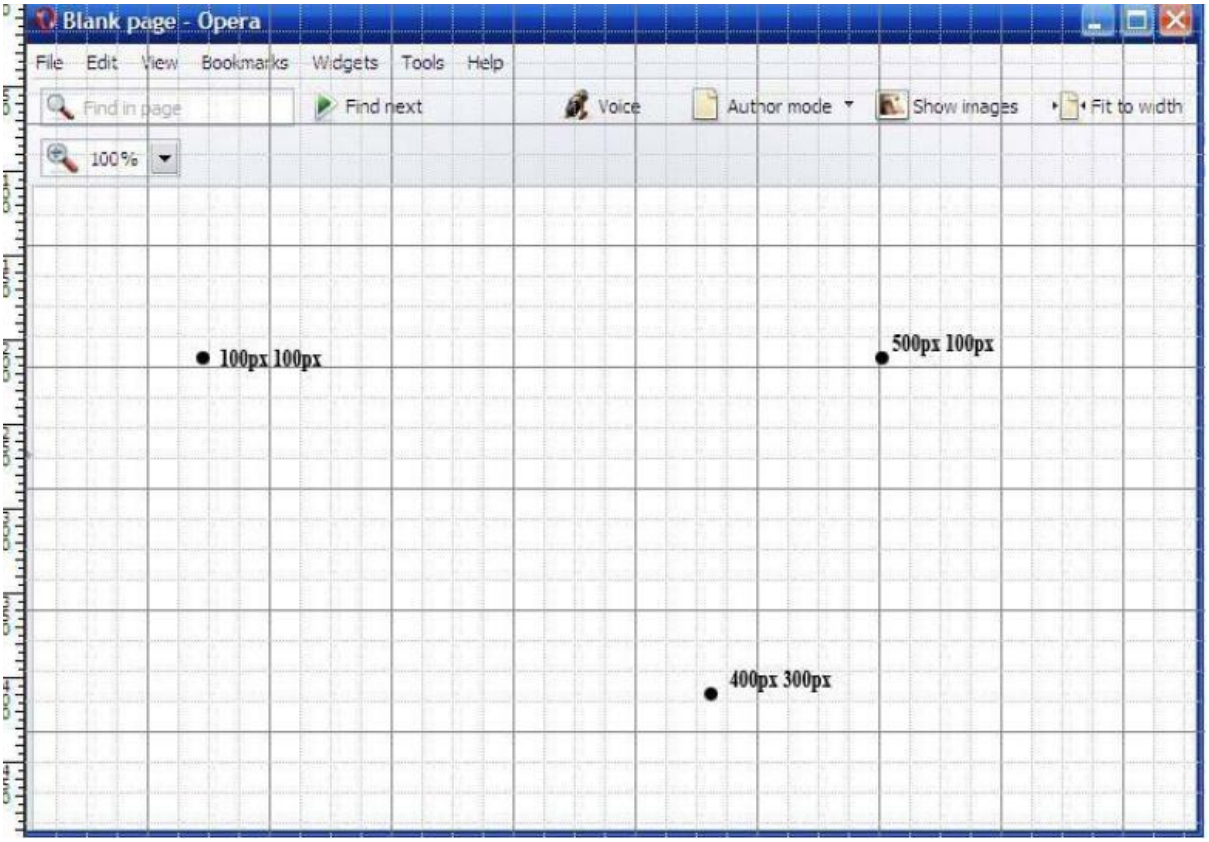
```
<!DOCTYPE html>
<html>
<head>
  <title>Xem Ví dụ</title>
  <meta charset="utf-8">
  <style type="text/css">
        .nguyen{color:blue;}
        .thanh{font-size:30px;}
        .nhan{background-color:yellow;}
  </style>
</head>
<body>
  <p class="nguyen">Tài liệu học Lập Trình Web 01</p>
  <p class="nguyen thanh">Tài liệu học Lập Trình Web 02</p>
  <p class="nguyen thanh nhan">Tài liệu học Lập Trình Web 03</p>
</body>
</html>
```

14. Position (Vị trí)

Thuộc tính **position** mang lại nhiều khả năng để tạo một cách trình bày tiên tiến và chính xác cho trang web.

Nguyên lý hoạt động của Position

Hãy tưởng tượng cửa sổ trình duyệt của bạn giống như một hệ tọa độ và với position bạn có thể đặt một đối tượng web ở bất cứ vị trí nào trên hệ tọa độ này.



Giả sử chúng ta muốn định vị một ảnh ở vị trí 70px cách đỉnh và 90px từ bên trái tài liệu, chúng ta sẽ viết CSS như sau:

```
img { position:absolute; top:70px; left:90px }
```

14.1. Static

position: static là giá trị mặc định của position. Dù chúng ta có khai báo chúng hay không thì các element sẽ được sắp xếp vị trí một cách như bình thường trên trang web.

Ví dụ:

Code HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Demo 1</title>
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="b1.css">

</head>
<body>
  <div class="vang"></div>
  <div class="xanh"></div>
</body>
</html>
```

Code CSS

```
.vang {
```

```
  width: 100px;
  height: 100px;
  background-color: #FED500;
}
.xanh {

  width: 100px;
  height: 100px;
  background-color: #1BA160;
  position: static;   // Khai báo "static"
}
```

Dù chúng ta có khai báo: position: static; // Khai báo "static" hay không thì vẫn có kết quả như sau:

14.2. Relative position (Định vị tương đối)

Relative: Định vị trí tuyệt đối cho các thành phần, không gây ảnh hưởng tới vị trí ban đầu hay các thành phần khác.

Với **position: relative** và các giá trị khác ngoài static, chúng ta có thể dễ dàng thay đổi vị trí của chúng.

Muốn di chuyển hình vuông màu vàng sang bên cạnh hình vuông màu xanh, chúng ta dùng CSS

Code CSS

```
.vang {
  position: relative; /*chúng ta có thể di chuyển được element*/
  width: 100px;
  height: 100px;
  top:100px; /*dịch chuyển xuống 100px từ vị trí ban đầu của nó*/
  left: 100px; /*dịch chuyển sang phải 100px*/
  background-color: #FED500;
}
.xanh {
```

```
  width: 100px;
  height: 100px;
  background-color: #1BA160;
  position: static;   // Khai báo "static"
}
```

Kết quả:

Note: Sử dụng `position: relative` cho một element thì sẽ không ảnh hưởng tới vị trí của các element khác.

14.3. Absolute position (Định vị tuyệt đối)

Với position: relative, một element được dịch chuyển tới một vị trí mới dựa trên ví trí bình thường của chính nó. Tuy nhiên, position: absolute sẽ dịch chuyển vị trí của nó tương ứng với thẻ cha của nó.

Một element được khai báo với thuộc tính position: absolute sẽ được loại bỏ khỏi luồng document (document flow). Vị trí mặc định của element sẽ là điểm bắt đầu (top-left) của element cha. Nếu nó không có bất cứ thẻ cha nào thì thẻ document <html> sẽ là cha của nó.

Vì position: absolute đã được loại bỏ khỏi document flow, các element khác do element được định nghĩa position: absolute được coi là đã bị xóa khỏi trang web.

Ví dụ 1:

Cũng với ví dụ trên, nhưng thêm 1 div .container bọc lấy 2 div trên

Code HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>Demo 1</title>
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="b2.css">

</head>
<body>
  <div class="container">
        <div class="vang"></div>
        <div class="xanh"></div>
```

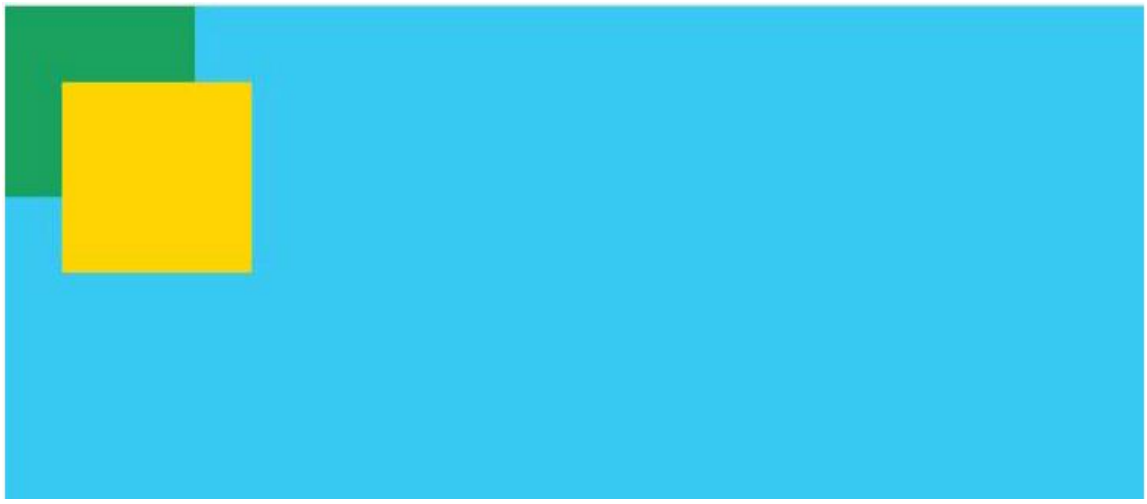
```
  </div>

</body>
</html>
```

Code CSS

```
*{margin: 0; padding: 0;}
.container{
  width: 600px;
  height: 300px;
  background-color: #3AC9F3;
}
.vang {
  position: absolute;
  width: 100px;
  height: 100px;
  background-color: #FED500;
  left: 30px;
  top:40px;
}
.xanh {
  width: 100px;
  height: 100px;
  background-color: #1BA160;
  }
```

Kết quả khi View



Nhận xét:

Div container (màu xanh biển) chứa div vang (màu vàng) và div xanh (màu xanh lá cây)

Vì div vang có position: absolute; nên nó sẽ nhận div container là div cha và canh trên nó sẽ cách cạnh trên của container 40px và cạnh trái nó cách cạnh trái container 30px

Ví dụ 2

Code HTML

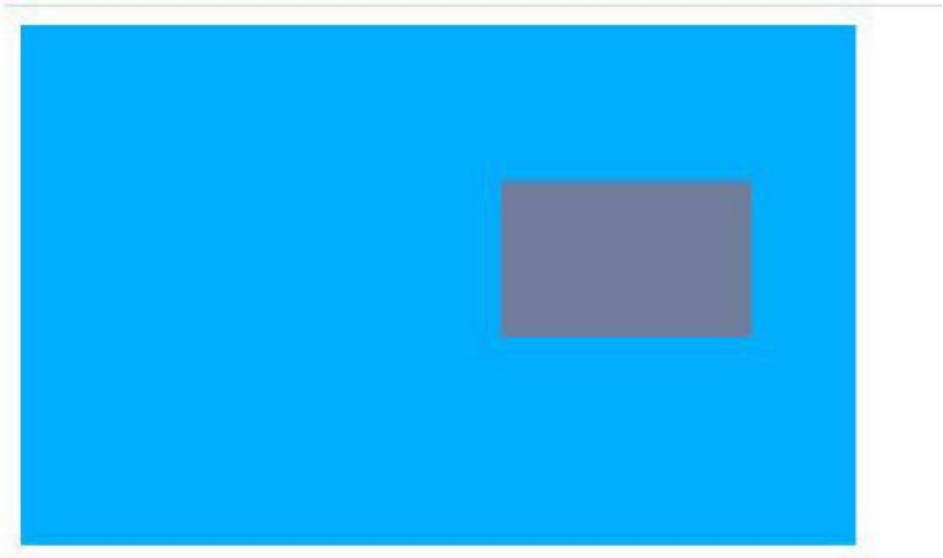
```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="Pos1.css">
    <title>Document</title>
</head>
<body>
    <div class="content">
        <div class="box"></div>
    </div>
</body>
</html>
<!-- Ví dụ về 1 phần tử relative và phần tử con absolution -->
```

Code CSS

```css
.content {
    position: relative;
    width: 400px;
    height: 250px;
    background-color: #00aefd;
    /* margin: 20px; */
  }

  .box {
    position: absolute;
    width: 30%;
    height: 30%;
    bottom: 100px;
    right: 50px;
    opacity: 0.5;
    background-color: #e74c3c;
  }
```

Khi View trên trình duyệt ta thấy:



Div màu xanh biển (content); div màu xám (box), nhìn vào hình ta thấy cạnh dưới hình chữ nhật màu xám sẽ cách cạnh dưới hình chữ nhật màu xanh 100px; cạnh phải của hình chữ nhật màu xám cách cạnh phải hình chữ nhật màu xanh 50px;

15. Thuộc tính float và Clear

15.1 Float

Thuộc tính Float sử dụng để chuyển một phần tử sang góc trái hoặc phải của không gian bao quanh nó, rất cần thiết trong việc định dạng bố cục trang.

Theo mặc định, tất cả các phần tử HTML là không float.

Thuộc tính Float có thể có một trong các giá trị sau:

- left: Cố định phần tử về bên trái.
- right: Cố định phần tử về bên phải.
- none: Nằm tại chính vị trí của nó (trạng thái bình thường).
- inherit (kế thừa): Phần tử kế thừa giá trị từ float cha.

Chú ý: Khi một thành phần được CSS float là left hoặc right thì tất cả các thẻ cùng cấp phía sau nó sẽ được tràn lên phía trên và lấp đầy chỗ trống của hàng chứa thẻ được CSS float left hoặc right.

Ví dụ 1:

```
<body>
    <p>Trong ví dụ này, hình ảnh sẽ nổi bên phải và văn bản sẽ bao quanh hình ảnh.</p>

    <p><img src="nho1.jfif" alt="Quả nho" style="width:170px;height:170px;
    margin-left:15px;">
    Nho là một loại quả mọng lấy từ các loài cây thân leo thuộc chi Nho (Vitis).
    Quả nho mọc thành chùm từ 6 đến 300 quả, chúng có màu đen, lam, vàng, lục, đỏ-tía hay trắng. Khi chín, quả nho có thể ăn tươi hoặc được sấy khô để làm nho khô, cũng như được dùng để sản xuất các loại rượu vang, thạch nho, nước quả, mật nho, dầu hạt nho. Trong tiếng Trung, nó được gọi là bồ đào và khi người ta nói đến rượu bồ đào tức là rượu sản xuất từ quả nho. Phần lớn nho được thu hoạch từ loài được trồng là Vitis vinifera, loài làm rượu vang châu Âu bản địa của vùng Địa Trung Hải và Trung Á. Một lượng nhỏ trái và rượu có nguồn gốc từ các loài châu Mỹ và châu Á. </p>
</body>
```

Code css

```
img { float: right;}
p{ background:#FFFACD;width: 636px;}
```

15.1 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.

Ở ví dụ trên, khi chúng ta float tấm ảnh qua trái thì mặc nhiên văn bản sẽ được tràn lên để lắp vào chỗ trống. Nhưng khi chúng ta đặt vào văn bản thuộc tính clear thì chúng ta có quyền quyết định xem phần văn bản đó có được tràn lên hay không.

BÀI 2: SỬ DUNG HTML VÀ CSS ĐỂ LÀM VIỆC VỚI ĐỐI TƯỢNG

Mục tiêu:

- Nắm được các kỹ năng làm việc với đối tượng hình ảnh, bảng biểu, form
- Nắm được cách kết nối giữa các đối tượng với nhau

Nội dung:

1. Làm việc với hình ảnh

1.1. Sử dụng HTML5 để chèn hình ảnh vào Web

Trong HTML, ảnh được định nghĩa với thẻ <img>.

Thẻ <img> là trống, nó chỉ chứa các thuộc tính, và không có thẻ đóng.

Các thuộc tính src xác định URL (địa chỉ web) của ảnh:

<img src="url" alt="some_text">

Thuộc tính src

Thuộc tính này chính là đường dẫn của các file ảnh, hoặc địa chỉ ảnh trên trang web

Ví dụ 1:

```
<img src="wrongname.gif" alt="HTML5 Icon" style="width:128px;height:128px;">
```

Ví dụ 2:

```
<img src=" https://photo-baomoi.zadn.vn/w700_r16x9/2021_04_08_114_38468552/093227660824e17ab835.jpg" alt="Youn Yuh-jung">
```

Thuộc tính alt

Thuộc tính alt định nghĩa một văn bản thay thế cho hình ảnh, nếu hình ảnh không thể hiện thị được.

Thuộc tính alt cung cấp thông tin thay thế cho một hình ảnh, nếu một người vì lí do nào đó không thể xem nó (vì kết nối mạng chậm, hoặc lỗi trong thuộc tính src, hoặc nếu người dùng sử dụng một trình đọc màn hình).

Nếu trình duyệt không thể tìm thấy ảnh, nó sẽ hiển thị nội dung chứa trong alt:

1.2. Định dạng hình ảnh bằng CSS3

1.2.1. Thuộc tính Width và Height

2 thuộc tính này dùng để xác định chiều ngang và chiều cao của ảnh

Ví dụ:

```
<img src="wrongname.gif" alt="HTML5 Icon" style="width:128px;height:128px;">
```

1.2.2. Thiết lập vị trí của hình ảnh so với văn bản

Trong trang web, việc thiết lập vị trí của hình ảnh so với văn bản nằm ở xung quanh cũng đóng một vai trò tương đối quan trọng (bởi vì nó mang tính chất thẩm mỹ cho trang web)

- Để thiết lập vị trí của tấm hình so với văn bản nằm ở xung quanh thì chúng ta thêm thuộc tính align vào bên trong thẻ <img> với cú pháp như sau:

```
align="value"
```

-Trong đó, value có thể được xác định dựa theo một trong năm loại giá trị:

left	- Tấm hình được đẩy sang bên trái, phần văn bản (được khai báo phía sau nó) sẽ nằm bên phải & bao xung quanh tấm hình.	
right	- Tấm hình được đẩy sang bên phải, phần văn bản (được khai báo phía sau nó) sẽ nằm bên trái & bao xung quanh tấm hình.	

top	- Dòng đầu tiên của phần văn bản (được khai báo phía sau tấm hình) sẽ nằm ở vị trí cao nhất so với tấm hình.	
middle	- Dòng đầu tiên của phần văn bản (được khai báo phía sau tấm hình) sẽ nằm ở vị trí trung bình (giữa) so với tấm hình.	
bottom	- Dòng đầu tiên của phần văn bản (được khai báo phía sau tấm hình) sẽ nằm ở vị trí thấp nhất so với tấm hình.	

2. Làm việc với bảng biểu

2.1. Sử dụng HTML5 để tạo bảng biểu

2.1.1. Các thẻ sử dụng để tạo bảng

Thông thường, để tạo được một cái bảng thì chúng ta cần phải sử dụng bốn loại thẻ:

- Thẻ <table> dùng để xác định một cái bảng.
- Thẻ <tr> dùng để xác định một hàng bên trong bảng.
- Thẻ <th> dùng để xác định một ô (tiêu đề) bên trong hàng.
- Thẻ <td> dùng để xác định một ô (bình thường) bên trong hàng.

Ví dụ: Bảng bên dưới có sáu hàng tương ứng với sáu phần tử <tr>, hàng đầu tiên có ba ô tiêu đề tương ứng với ba phần tử <th>, năm hàng còn lại gồm mười lăm ô bình thường tương ứng với mười lăm phần tử <td>

Họ tên	Ngày sinh	Giới tính
Trần Anh Đức	03/08/1993	Nam
Kiều Thị Thu Hằng	04/09/1991	Nữ
Vương Thị Lê Na	06/10/1991	Nữ
Dương Kim Thương	16/11/1990	Nam
Mai Đức Hiếu	18/06/1989	Nam

Lưu ý: Ô tiêu đề là loại ô mà văn bản nằm bên trong nó mặc định được tô đậm & canh giữa.

2.1.2. Các bước để tạo bảng

- Để tạo một cái bảng thì chúng ta nên thực hiện lần lượt các bước như sau:

- Bước 1: Xác định một cái bảng.
- Bước 2: Xác định số hàng nằm bên trong bảng.
- Bước 3: Xác định số ô nằm bên trong mỗi hàng.
- Bước 4: Xác định nội dung của từng ô.
- Bước 5: Thiết lập thuộc tính border với giá trị 1 để tạo đường viền cho bảng và các ô.

Ví dụ:

```
<table border="1">
    <tr>
      <th>Họ tên</th>
      <th>Ngày sinh</th>
```

```
        <th>Giới tính</th>
    </tr>
    <tr>
        <td>Phan Minh Tiến</td>
        <td>03/08/1993</td>
        <td>Nam</td>
    </tr>
    <tr>
        <td>Lê Thị Thu Hằng</td>
        <td>04/09/1991</td>
        <td>Nữ</td>
    </tr>
    <tr>
        <td>Vương Phương Hoa</td>
        <td>06/10/1991</td>
        <td>Nữ</td>
    </tr>
</table>
```

2.2. Sử dụng CSS3 để định dạng bảng biểu

Để định dạng bảng chúng ta dùng một số thuộc tính sau:

- border: Thiết lập độ dày của cái đường viền bao xung quanh bảng và các ô.
- cellspacing: Thiết lập khoảng cách nằm giữa mỗi hai đường viền lân cận.

```
<table border="1" cellspacing="0" cellpadding="5" > //Đường viền còn có 1 nét vì khoảng cách 2 đường viền =0
```

- cellpadding: Thiết lập khoảng cách vùng đệm bên trong các ô.
- bgcolor: Thiết lập màu nền cho bảng hoặc các ô.
    - *Nếu muốn thiết lập màu nền cho nguyên cái bảng thì ta đặt thuộc tính bgcolor nằm bên trong thẻ <table>*
    - *Nếu muốn thiết lập màu nền cho một hàng thì ta đặt thuộc tính bgcolor nằm bên trong thẻ <tr>*
    - *Nếu muốn thiết lập màu nền cho một ô thì ta đặt thuộc tính bgcolor nằm bên trong thẻ <th> hoặc <td>*
- Width: Thiết lập chiều rộng cho bảng
    - *Nếu muốn thiết lập chiều rộng cho bảng thì ta đặt thuộc tính width nằm bên trong thẻ <table>*
    - *Nếu muốn thiết lập chiều rộng cho một ô thì ta đặt thuộc tính width nằm bên trong thẻ <th> hoặc <td>*

Ví dụ:

```
<table border="1" width="550">
    <tr width="50%">
        <th>Họ tên</th>
        <th>Ngày sinh</th>
        <th>Giới tính</th>
    </tr>
```

```
    <tr>
        <td>Trần Anh Đức</td>
        <td>03/08/1993</td>
        <td>Nam</td>
    </tr>
    <tr>
        <td>Kiều Thị Thu Hằng</td>
        <td>04/09/1991</td>
        <td>Nữ</td>
    </tr>
    <tr>
        <td>Vương Thị Lê Na</td>
        <td>06/10/1991</td>
        <td>Nữ</td>
    </tr>
</table>
```

- Height: Thiết lập chiều cao cho bảng

- Nếu muốn thiết lập chiều cao cho bảng thì ta đặt thuộc tính height nằm bên trong thẻ <table>

- Nếu muốn thiết lập chiều cao cho một ô thì ta đặt thuộc tính height nằm bên trong thẻ <th> hoặc <td>

Ví dụ:

```
<table border="1" height="400">
    <tr height="75%">
        <th>Họ tên</th>
        <th>Ngày sinh</th>
        <th>Giới tính</th>
    </tr>
    <tr>
        <td>Trần Anh Đức</td>
        <td>03/08/1993</td>
        <td>Nam</td>
    </tr>
    <tr>
        <td>Kiều Thị Thu Hằng</td>
        <td>04/09/1991</td>
        <td>Nữ</td>
    </tr>
    <tr>
        <td>Vương Thị Lê Na</td>
        <td>06/10/1991</td>
```

```
    <td>Nữ</td>
  </tr>
</table>
```

- align: Canh lề cho nội dung bên trong ô (theo chiều ngang)

+Thuộc tính align có thể nhận một trong bốn giá trị sau đây: left, right, center, justify

Ví dụ:

```
<table border="1" width="1000">
  <tr>
    <th width="25%">Cột 1</th>
    <th width="25%">Cột 2</th>
    <th width="25%">Cột 3</th>
    <th width="25%">Cột 4</th>
  </tr>
  <tr>
    <td align="left">X</td>
    <td align="center">X</td>
    <td align="right">X</td>
    <td align="justify">X</td>
  </tr>
</table>
```

- valign: Canh lề cho nội dung bên trong ô (theo chiều dọc)

+Thuộc tính valign có thể nhận một trong ba giá trị sau đây: top, middle (mặc định), bottom

Ví dụ:

```
<table border="1" height="400">
  <tr>
    <td valign="top">X</td>
    <td valign="middle">X</td>
    <td valign="bottom">X</td>
  </tr>
</table>
```

3. Làm việc với Form

3.1. Cách tạo form và nút điều khiển

Form là 1 đối tượng dùng để giao tiếp người và máy tính, chúng ta dùng nó để nhập thông tin vào máy tính để lưu trữ hoặc xử lý một thao tác khác. Có rất nhiều cách tạo form, chúng ta sẽ tham khảo lần lượt qua các ví dụ.

Ví dụ 1a:

```
<body>
    <form method="post" action="">
     Username: <input type="text" value="" name="username" id="username"/> <br/>
```

```
    Password: <input type="text" value="" name="password" id="password"/> <br/>
    Re Password: <input type="text" value="" name="re-password" id="re-password"/> <br/>
    <input type="submit" value="Register" />
   </form>
  </body>
```

Trong form thẻ Input là thẻ được dùng để tạo các điều khiển nhập liệu trong một form, nó có rất nhiều dạng.

3.1.1 <input type="text">

Kiểu này được dùng để nhập liệu cho dạng ô văn bản. Lưu ý độ rộng mặc định của trường văn bản là 20 kí tự

3.1.2. <input type="password">

Kiểu này dùng để định nghĩa một trường mật khẩu, Các kí tự trong trường password được ẩn đi (nó được mã hóa hiển thị thành hình sao hoặc các chấm tròn).

3.1.3. <input type="submit">

Kiểu này dùng định nghĩa một nút để gửi dữ liệu từ form người sử dụng nhập tới nơi xử lý dữ liệu của form này (form-handler).

Form-handler thường mà một trang chạy ở phía server, được viết bằng các ngôn ngữ ở phía máy chủ như PHP, ASP.NET, JSP … cho xử lý dữ liệu đầu vào.

Form-handler được chỉ định bởi các thuộc tính "action" trong form đó:

Ví dụ:

```
<body>
<form action="action_page.php">
Họ đệm:<br>
<input type="text" name="txtHoDem" value="Mickey">
<br>
Tên:<br>
<input type="text" name="txtTen" value="Mouse">
<br><br>
<input type="submit" value="Submit">
</form>
<p>Nếu bạn click vào nút "Submit", dữ liệu trong form sẽ được gửi tới
 một trang có tên gọi là "action_page.php".</p>
</body>
```

3.1.4. <input type="radio">

Định nghĩa một nút radio. Các nút Radio cho phép người sử dụng chọn duy nhất một lựa chọn trong danh sách các lựa chọn:

Ví dụ

```
<body>
    <form action="action_page.php">
        <input type="radio" name="gender" value="male" checked> Male<br>
        <input type="radio" name="gender" value="female"> Female<br>
```

```
            <input type="radio" name="gender" value="other"> Other<br><br>
            <input type="submit">
        </form>
</body>
```

3.1.5. <input type="checkbox">

Định nghĩa một nút checkbox. Checkboxes cho phép người sử dụng chọn không hoặc nhiều lựa chọn trong các lựa chọn đưa ra .

```
<body>
        <form action="action_page.php">
            <input type="checkbox" name="vehicle1" value="Xe đạp">Tôi có một chiếc xe đạp
            <br>
            <input type="checkbox" name="vehicle2" value="Ô tô">Tôi có một chiếc ô tô
            <br><br>
            <input type="submit">
        </form>
</body>
```

3.1.6. <input type="button">

Định nghĩa 1 nút thông thường (không phải là nút gửi dữ liệu đến 1 trang)

```
<input type="button" onclick="alert('Website Tìm ở đây chào bạn!')" value="Click Tôi!">
```

Đến version HTML5 đã có bổ sung thêm một số kiểu nút điều khiển mới như sau:

Sau đây là một số kiểu dữ liệu đầu vào mới bổ sung trong HTML5

- color
- date
- datetime
- datetime-local
- email
- month
- number
- range
- search
- tel
- time
- url
- week

3.1.7. <input type="number">

Được sử dụng cho trường dữ liệu đầu vào có chứa các giá trị là số, nó tạo ra một điều khiển dạng Spiner . Chúng ta có thể cài đặt các giới hạn trong kiểu đầu vào này. Tùy thuộc vào sự hỗ trợ của các trình duyệt mà các giới hạn này có thể áp dụng cho các trường dữ liệu đầu vào.

```
<body>
    <p>
        Phụ thuộc vào sự hỗ trợ của các trình duyệt:<br>
        Các giới hạn Numeric sẽ áp dụng cho các trường dữ liệu đầu vào.
    </p>
    <form action="action_page.php">
        Chọn giá trị (trong khoảng từ 1 tới 5):
        <input type="number" name="quantity" min="1" max="5">
        <input type="submit">
    </form>
    <p><b>Chú ý:</b>type="number"không được hỗ trợ trong IE9 và trước đó.</p>
</body>
```

3.1.8. <input type="date">

Tạo một điều khiển nhập dữ liệu ngày

```
<input type="date" name="bday">
```

3.1.9. <input type="email">

Được sử dụng cho các trường dữ liệu đầu vào có chứa một địa chỉ e-mail. Phụ thuộc vào sự hỗ trợ của các trình duyệt, địa chỉ e-mail có thể được tự động xác nhận khi gửi đến. Một vài smartphones có thể nhận ra các loại email, và bổ sung thêm ".com" vào bàn phím để phù hợp với dữ liệu đầu vào của email.

```
<body>
    <form action="action_page.php">
        E-mail:
        <input type="email" name="email">
        <input type="submit">
    </form>
    <p><b>Note:</b>type="email" không được hỗ trợ bởi IE9 và các phiên bản trước đó.</p>
</body>
```

3.1.10. Thẻ Textarea

Thẻ HTML <textarea> được sử dụng để định nghĩa một ô nhập văn bản nhiều dòng. Có thể nhập đoạn văn bản dài vô hạn và hiển thị với độ dài ô nhập là cố định.

Kích thước của HTML textarea được định nghĩa bởi các thuộc tính <cols> và <rows> hoặc có thể được định nghĩa thông qua tính chất height và width của CSS.

Các thuộc tính mới của Textarea trong HTML 5

Thuộc tính	Mô tả
autofocus	Chỉ định rằng một textarea sẽ được tự động focus khi page được tải xong.
form	Chỉ định textarea thuộc về một hoặc nhiều form.
maxlength	Chỉ định số ký tự tối đa mà bạn có thể nhập vào textarea.
placeholder	Chỉ định một gợi ý ngắn mô tả giá trị dự kiến của một textarea.

required	Chỉ định textarea phải nhập giá trị vào.
wrap	Chỉ định rằng các văn bản trong vùng văn bản được bao bọc như thế nào vào thời điểm submit form.

Ví dụ 1

```
<textarea rows="9" cols="70">
    Đây là một ví dụ sử dụng thẻ textarea với thuộc tính rows và cols
</textarea>
```

Ví dụ 2

```
<body>
    <form action="#" id="usrform">
        Name: <input type="text" name="usrname">
        <input type="submit">
    </form>
    <br>
    <textarea rows="9" cols="70" name="comment" form="usrform">
        Nhập văn bản...
    </textarea>
</body>
```

3.2. Định dạng form

Form trong HTML thường được sử dụng để thu thập dữ liệu đầu vào của người sử dụng. Phần tử <form> định nghĩa một Form trong HTML. Trong form có thể chứa các các kiểu phần tử khác nhau như các ô nhập dữ liệu (textboxes), các ô cho người dùng lựa chọn (checkboxes hoặc radio buttons), các nút cho người dùng kích gửi dữ liệu (submit buttons) và nhiều phần tử khác mà chúng ta đã trình bày ở phần trên.

Trong mỗi form có nhiều thuộc tính đi kèm khi chúng ta khai báo thẻ form

3.2.1. Thuộc tính Action

Thuộc tính action xác định hành động được thực hiện khi form được gửi đi khi người sử dụng kích nút submit. Cách phổ biến nhất khi gửi dữ liệu của form tới server là sử dụng một nút submit. Thông thường form được gửi tới một trang web chạy trên máy chủ web.

```
<form action="action_page.php">
```

Nếu thuộc tính action bỏ qua, action sẽ được thiết lập tới trang hiện tại.

3.2.2. Thuộc tính Method

Thuộc tính method xác định kiểu phương thức HTTP (GET hoặc POST) được sử dụng gửi dữ liệu trên form:

```
<form action="action_page.php" method="get">
<form action="action_page.php" method="post">
```

Sử dụng GET khi nào?

Chúng ta có thể sử dụng GET (mặc định phương thức này). Sử dụng GET nếu việc gửi form là thụ động (giống như bạn thực hiện truy vấn trên các máy tìm kiếm) và dữ liệu không cần mã hoá, không chứa các thông tin nhạy cảm như mật khẩu, v.v. Khi bạn sử dụng GET, dữ liệu form sẽ bị nhìn thấy trên thanh địa chỉ của trang như ví dụ dưới:

Ví dụ

```
<form action="action_page.php" method="get">
```

```
        Họ đệm:<br>
        <input type="text" name="txtHoDem" value="Trần Hùng"><br>
        Tên:<br>
        <input type="text" name="txtTen" value="Dũng"><br>
        <input type="submit" value="Submit">
</form>
```

Nếu bạn chạy code và bấm vào nút Submit thì sẽ thấy trên thanh địa chỉ nội dung

action_page.php?txtHoDem=Trần+Hùng&txtTen=Dũng

Sử dụng POST khi nào?

Chúng ta nên sử dụng POST: Trong trường hợp nếu form cập nhật dữ liệu hoặc dữ liệu trên form gửi đi bao gồm các thông tin nhạy cảm như mật khẩu, mã thẻ ngân hàng,v.v. POST cung cấp cơ chế bảo mật hơn bởi vì dữ liệu được gửi đi không được hiển thị trên thanh địa chỉ của trang.

3.2.3. Thuộc tính Name

Để lấy được dữ liệu đúng khi gửi đi, mỗi trường phải có một thuộc tính name. Ví dụ này sẽ chỉ gửi trường "Họ đệm", theo bạn tại sao lại không gửi trường "Tên"?:

```
<form action="action_page.php" method="get">
        Họ đệm:<br>
        <input type="text" name="txtHoDem" value="Trần Hùng"><br>
        Tên:<br>
        <input type="text" value="Dũng"><br>
        <input type="submit" value="Submit">
</form>
```

3.2.4. Nhóm dữ liệu trong Form với <fieldset>

Phần từ <fieldset> nhóm dữ liệu liên quan trong một form. Phần tử <legend> xác định một phụ đề cho phần tử <fieldset>.

Ví dụ

```
<form action="action_page.php">
        <fieldset style="width: 300px;">
                <legend>Thông tin cá nhân:</legend>
                Họ đệm:<br>
                <input type="text" name="firstname" value="Hùng"><br>
                Tên:<br>
                <input type="text" name="lastname" value="Phạm Văn"><br><br>
                <input type="submit" value="Submit">
        </fieldset>
    </form>
```

BÀI 3: JAVASCRIPT VÀ JQUERY

1. Giới thiệu về Javascript

JavaScript là một ngôn ngữ gia thêm khả năng tương tác cho website của bạn (ví dụ: trò chơi, các phản hồi khi các nút được nhấn hoặc nhập dữ liệu trên form, kiểu động, hoạt họa).

JavaScript (viết tắt là "js") là một ngôn ngữ lập trình mang đầy đủ tính năng của một ngôn ngữ lập trình động mà khi nó được áp dụng vào một tài liệu HTML, nó có thể đem lại khả năng tương tác động trên các trang web. Cha đẻ của ngôn ngữ này là Brendan Eich, đồng sáng lập dự án Mozilla, quỹ Mozilla, và tập đoàn Mozilla.

JavaScript thật sự rất linh hoạt. Bạn có thể bắt đầu với các bước nhỏ, với thư viện ảnh, bố cục có tính thay đổi và phản hồi đến các nút nhấn. Khi có nhiều kinh nghiệm hơn, bạn có thể tạo ra các trò chơi, hoạt họa 2D hoặc 3D, ứng dụng cơ sở dữ liệu toàn diện và nhiều thứ khác!

Bản thân Javascript là một ngôn ngữ linh động. Các nhà phát triển đã viết ra một số lượng lớn các công cụ thuộc top của core Javascript, mở ra một lượng lớn tính năng bổ sung với ít nỗ lực nhất. Nó bao gồm:

Giao diện lập trình ứng dụng trên trình duyệt (API) — Các API được xây dựng bên trong các trình duyệt web, cung cấp tính năng như tạo HTML động, cài đặt CSS, thu tập và điều khiển video trực tiếp từ webcam của người dùng hoặc sinh ra đồ dọa 3D và các mẫu audio.

Các API bên thứ ba cho phép nhà phát triển kết hợp tính năng trong website của họ từ người cung cấp nội dung khác chẳng hạn như Twitter hay Facebook.

Từ các framework và thư viện bên thứ ba bạn có thể áp dụng tới tài liệu HTML của bạn, cho phép bạn nhanh chóng xây dựng được các trang web và các ứng dụng.

1.1. Cách sử dụng JavaScript

1.1.1. Sử dụng Javascript trực tiếp trên file HTML

-Để có thể viết được HTML trên file HTML thì bắt buộc chúng ta cần phải khai báo bắt buộc theo cú pháp:

```
<script type="text/javascript">
        //code javascript ở đây
  </script>
```

đoạn script này có thể đặt ở bất cứ đâu trong file HTML.

1.1.2. Import javascript từ một file bên ngoài

Tương tự như css, javascript cũng có thể import từ một file bên ngoài vào được và bắt buộc file đó phải có phần mở rộng .js theo cú pháp:

```
<script type="text/javascript" src="path/tên file.js"></script>
```

Lưu ý: Chúng ta nên đặt dòng lệnh này ở cuối phần body

1.1.3. Viết javascript trong thẻ HTML

-Cách này cũng khá phổ biến khi muốn gọi một hàm hay thực thi một đoạn code javascript ngắn.

Cú pháp:

```
<button type="button" onclick="code">Ấn vào rồi biết</button>
```

-Trong đó: **code** là nơi chứa codejavascript.

Ví dụ: Trong file HTML chúng ta tạo 1 nút

```
<body >
        <h3>Demo JavaScript</h3>
        <button onclick="test()">Click vào đây</button>

        <script type="text/javascript" src="1.js"></script>
</body>
</html>
```

Trong file js chúng ta có đoạn code

```
var a=5;
var b=7;
console.log(a);
function test(){
    console.log(a+b);
}
```

2. Nhập môn Javascript

2.1. Hằng, biến và các kiểu dữ liệu trong Javascript

2.1.1. Hằng

Hằng (Constant) là một giá trị không đổi trong bất kỳ 1 ngôn ngữ lập trình nào và JavaScript cũng vậy. Để khai báo 1 hằng chúng ta sử dụng cú pháp:

const tenHang = giatri;

Trong đó:

tenHang: Là tên của hằng các bạn muốn đặt.

giatri: Là giá trị của hằng, có thể là số, chuỗi, mảng, object.

Lưu ý: Tên hằng phải được đặt theo quy tắc sau đây

- Tên hằng bắt đầu phải là chữ cái hoặc ký tự '_'
- Tên hằng không được bắt đầu bằng số.
- Tên hằng có độ dài không giới hạn.

Ví dụ:

```
const MyConst = 4; // đúng
    const _MyConst = 4; // đúng
    const __MyConst = 4; // đúng
    const MyConst1 = 4; // đúng
    const _MyConst1 = 4; // đúng
    const __MyConst1 = 4; // đúng
    const 90MyConst1 = 4; // sai
    const -MyConst1 = 4; // sai
```

2.1.2. Biến

Trong javascript chúng ta có thể khai báo biến bằng các cách sau đây:

Cách 1: Sử dụng từ khóa var.

var tenBien = giaTri;

Cách 2: Không cần sử dụng từ khóa.

tenBien = giaTri

Trong đó:

- **tenBien**: Là tên của biến các bạn muốn đặt.
- **giaTri**: Là giá trị của biến, có thể là số,chuỗi,mảng,object.

Chú ý: Tên Biến phải được đặt theo quy tắc sau đây

- Tên biến bắt đầu phải là chữ cái hoặc ký tự '_'
- Tên biến không được bắt đầu bằng số.
- Tên biến có độ dài không giới hạn.

Ví dụ:

```
var MyVariable = 4; // đúng
var _MyVariable = 4; // đúng
var __MyVariable = 4; // đúng
var MyVariable1 = 4; // đúng
var _MyVariable1 = 4; // đúng
var __MyVariable1 = 4; // đúng
var 90MyVariable1 = 4; // sai
var -MyVariable1 = 4; // sai
```

2.1.3. Kiểu dữ liệu

- Cũng giống như trong PHP biến và hằng trong javascript cũng tự detect kiểu dữ liệu của biến và hằng.

Ví dụ:

```
var a = 5; //dữ liệu kiểu int
var a = 5.65; //dữ liệu kiểu float
var a = 'Toidicode'; //dữ liệu kiểu string
var a = [1, 2, 3, 5, 9]; //dữ liệu kiểu array
var a = new Array(1, 2, 3, 5, 9); //dữ liệu kiểu array
var a = {'b': 4, 'c': 5}; //dữ liệu kiểu object
```

2.1.4. Hiển thị nội dung ra trình duyệt

Trong javascript có thể hiển thị nội dung ra trình duyệt bằng rất nhiều cách, có 2 cách thông dụng nhất để các chúng ta có thể kiểm tra được dữ liệu.

Cách 1: dùng document.write() hoặc document.writeln().

Ví dụ: trong file .js chúng ta có code sau

```
var a = 'Hello World';
document.write(a);
```

Lưu ý: Cách này không thể in ra được thông tin của object.

Cách 2: dùng console.log().

Ví dụ: trong file .js chúng ta có code sau

```
var bien1= 12;
var bien2=10;
var bien3= bien1+bien2;
```

```
console.log(bien1);
console.log(bien2);
console.log(bien3);
```

Cách này có thể xuất được nội dung của tất cả các dạng dữ liệu và để xem được hiển thị của nó thì các bạn cần phải bật cửa sổ console của trình duyệt (thường ấn F12).

2.2. Toán tử và biểu thức

2.2.1. Toán tử

a. Toán tử gán.

-Toán tử gán trong Javascript để gán giá trị cho biến hoặc hằng,.. chúng ta sử dụng dấu =

```
var bien1= 12;
```

b. Toán tử số học

-Toán tử số học thực ra là các dạng tính toán như cộng, trừ, nhân, chia , danh sách các toán tử số học trong javascript hỗ trợ.

Cộng

-Phép cộng sẽ cộng 2 con số lại nếu nó là số, và sẽ nối chuỗi lại nếu là chuỗi.

Ví dụ:

```
var a = 6;
var b = 5;
var c = 'vu thanh ';
var d ='tai';
document.write(a + b); // 11
document.write('<br/>'); // xuống dòng cho dễ đọc
document.write(c + d); // vu thanh tai
```

Trừ

-Phép trừ chỉ dùng được với số

Ví dụ:

```
var a = 6;
var b = 5;
document.write(a - b); // 1
```

Nhân

-Phép nhân cũng chỉ dùng được với số.

Ví dụ:

```
var a = 6;
var b = 5;
document.write(a * b); // 30
```

Chia

-Phép chia dùng được với số.

Ví dụ:

```
var a = 10;
var b = 5;
document.write(a / b); // 2
```

Chia lấy dư

-Phép chia này sẽ lấy phần dư của phép chia

Ví dụ:

```
var a = 6;
var b = 5;
document.write(a % b); // 1
```

c. Toán tử kết hợp

-Toán tử kết hợp thực ra là một cách rút gọn của các toán tử số học.

#	Ví dụ	Chú thích
++	a++	Bằng a = a+1;
--	a--	Bằng a = a-1;
*=	a*=b	Bằng a = a*b;
/=	a/=b	Bằng a = a/b;
+=	a+=b	Bằng a = a+b;
-=	a-=b	Bằng a = a-b;
%=	a%=b	Bằng a = a%/b;

d. Toán tử quan hệ

Toán tử	Chú Thích
>	Lớn hơn
>=	Lớn hơn hoặc bằng
<	Nhỏ hơn
<=	Nhở hơn hoặc bằng
==	Bằng
!=	Khác (Không bằng)

e. Toán tử logic

&& (phép AND)

|| (phép OR)

! (phép NOT)

2.2.2. Biểu thức

Biểu thức là 1 tổ hợp các toán tử và các toán hạng, trong đó:

Toán tử: là các phép toán được sử dụng trong Javascript

Toán hạng: có thể là hằng, biến, hàm

2.3. Biểu thức điều kiện

2.3.1. Câu lệnh if

Cú pháp:

```
if (condition) {
  //nếu điều kiện đúng thì thực hiện
}
```

Trong đó: condition là một hoặc nhiều mệnh đề điều kiện có giá trị trả về TRUE/FALSE.

Ví dụ:

```
var a = 5;
var b = 6;
if (a != b) {
  //Vì a khác b nên code phía trong if sẽ được chạy
    document.writeln('a khác b');
    console.log('a khác b')
}
```

2.3.2. Câu lệnh if .. else

Cú pháp:

```
if (condition) { //nếu điều kiện đúng thì chạy code trong này }
 else { //nếu điều kiện sai thì chạy code trong này }
```

Ví dụ:

```
var a = 5;
var b = 6;
if(a>b) {document.write('a lớn hơn b');}
  else {document.write('a nhỏ thua b');}
```

2.3.3. Câu lệnh if…else lồng nhau

Ví dụ 1

```
var diem = 8.6;
if(diem<=5){document.write('Yếu');} else
   if(diem<=7) {document.write('Trung bình');} else
     if(diem<=8.5) {document.write('Khá');} else {document.write('Giỏi');}
```

Ví dụ 2

```
var a=6;b=8;c=2;

if(!(a>7 || b<=8)) {document.write("OK");} else {document.write("Not OK");}
```

2.3.4. Câu lệnh if-else rút gọn

Chúng ta cũng có thể rút gọn câu lệnh if-else với cú pháp như sau:

(dieukien) ? (đúng) : (sai)

Ví dụ:

```
var diem = 5;
diem == 5 ? document.write('Điểm Bằng 5') : document.write('Điểm khác 5');
```

2.3.5. Câu lệnh Switch case

Đây là một loại câu lệnh rẽ nhánh(hay còn gọi là câu lệnh điều kiện) có đặc điểm là để giải quyết các bài toán mà có các nhánh là các điều kiện cố định.

Cú pháp

```
switch (condition) {
   case value1:
      //code
      break;
   case value2:
      //code
      break;
   default:
      //code
      break;
}
```

Trong đó:

condition là biến muốn kiểm tra để dẽ nhánh.

value1,**value2**,.. là các giá trị tương ứng của condition mà các bạn muốn rẽ nhánh.

default là giá trị khác đối với tất cả các value trên(giống với else).

break dừng

Ví dụ:

```
var so = 5;

switch (so) {
   case 0:
      document.write('không');
      break;
   case 1:
      document.write('Một');
      break;
   case 2:
      document.write('Hai');
      break;
   case 3:
      document.write('Ba');
      break;
   case 4:
```

```
        document.write('Bốn');
        break;
    case 5:
        document.write('Năm');
        break;
    default:
        document.write('Không thỏa mãn');
        break;
}
```

2.4. Vòng lặp

Vòng lặp là một chuỗi sự kiện, hành động được lặp đi lặp lại theo một nguyên tắc nhất định. Ví dụ như vong lặp của con người: sinh-lão-bệnh-tử.

Và đối với Javascript nó hỗ trợ chúng ta 3 kiểu vòng lặp(loop) là for, forEach, while và do-while.

2.4.1. Vòng lặp for

Cú pháp

```
for (bienkhoitao; dieukienthucthi ; buocnhay) {
  # code...
}
```

Trong đó:

- bienkhoitao: là giá trị khởi tạo ban đầu của vòng lặp.
- dieukienthucthi: là điều kiện mà vòng lặp được phép chạy(chú ý: Nếu bạn muốn lặp vô tận thì có thể bỏ trống).
- buocnhay: là khoảng đệm nhảy của mỗi vòng lặp.

Ví dụ: chúng ta viết đoạn code này trong file .js

```
for (var i = 1; i <= 10; i++) {
    document.write('Dòng số: ' + i + '<br/>');
}
```

Hoặc nếu không viết trong file .js, thì viết trực tiếp trong file .html như sau:

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Ví dụ vòng for</title>
</head>
<body>
    <script type="text/javascript">
        for(var i=1;i<=10;i++){
            document.write('Dòng số: '+ i + '<br/>');
        }
```

```
    </script>
</body>
</html>
```

Công dụng của đoạn code này sẽ in ra các con số từ 1 đến 10 (xuống dòng mỗi số)

Lưu ý: *để trở thành vòng lặp vô tận thì chỉ cần để trống biểu thức điều kiện thứ 2 trong vòng lặp for.*

Ví dụ:

```
for (var i = 0; ; i++) {
   //code
}
```

2.4.2. Vòng lặp do-while

Vòng lặp do-while (thực hiện cái gì đó trong khi còn thoả điều kiện), Đây là một dạng vòng lặp cũng khá phổ biến trong javascript, đặc trưng của vòng lặp này là để lặp các dữ liệu mà không xác đinh được điều kiện dừng chính xác hoặc điều kiện dừng phức tạp và nó **thực thi câu lệnh trước rồi mới kiểm tra điều kiện**.

Cú pháp

```
do {
   // code
} while (condition);
```

Trong đó: **condition** là điều kiện để dừng vòng lặp, nếu bằng false thì vòng lặp sẽ dừng và ngược lại true vòng lặp sẽ chạy tiếp.

Ví dụ:

```
var i=2;
do{
     document.write(i+ '<br>');
      i++;
  } while(i<10);
```

2.4.3. Vòng lặp while

Vòng lặp while cũng giống như vòng lặp do-while là dùng để lặp các dữ liệu không biết chính xác điều kiện dừng hoặc điều kiện dừng phức tạp. Và vòng lặp while khác với vòng lặp do-while là nó sẽ tiến hành kiểm tra điều kiện xong rồi mới tiến hành lặp.

Cú pháp:

```
while (condition) {
          //code
  }
```

Trong đó: **condition** là điều kiện để dừng vòng lặp, nếu bằng false thì vòng lặp sẽ dừng và ngược lại true vòng lặp sẽ chạy tiếp.

Ví dụ:

```
//khởi tạo biến i=1
var i = 1;
//xác định điều kiện i>10 thì sẽ dừng vòng lặp
while(i <= 10){
  document.write(i);
  //tiến hành tăng i thêm 1 đơn vị
```

```
  i++;
}
```

2.4.4. Vòng lặp lồng nhau

Cũng giống như câu lệnh điều kiện thì vòng lặp cũng hoàn toàn có thể lồng nhau.

VD1: Vòng lặp for lồng nhau.

```
for(i = 0; i <= 10; i++) {
    for(j = i ; j <= 10; j++) {
                document.write('*');
            }
     document.write('<br>');
   }
```

VD2: Vòng lặp do-while lồng nhau.

```
var i = 0;
do {
    var j = i;
    do {
        document.write('*');
        j++;
        } while (j <= 10);
   document.write('<br>');
   i++;
  } while (i <= 10);
```

VD: Vòng lặp while lồng nhau.

```
var i = 0;
while (i <= 10) {
   var j = i;
   while (j <= 10) {
      document.write('*');
      j++;
   }
   document.write('<br>');
   i++;
}
```

2.5. Làm việc với dữ liệu kiểu mảng trong Javascript

What

- Khi chúng ta muốn lưu nhiều giá trị trong 1 biến phải dùng mãng (Array)

- Mãng là 1 loại biến đặc biệt, các biến bình thường chỉ lưu được 1 giá trị, mãng có thể lưu nhiều giá trị
- Khi lưu thành mãng thì tất cả các giá trị đó đều cùng 1 tên biến, chỉ khác là có thêm chỉ số phần tử của biến, vậy để truy cập vào từng phần tử của mãng chúng ta phải dùng tên biến kèm theo chỉ số phần tử.

Why

- Do chúng ta cần phải lưu nhiều giá trị cùng loại dữ liệu, do đó cần phải dùng mãng

How

2.5.1. Cách khai báo mãng

Ở trong javascript chúng ta có thể khai báo mảng bằng 2 cách sau đây:

Cách 1: Khai báo bằng dấu []

Cú pháp

var arr = [value1, value2, ..., valuen];

Trong đó

- **arr** là tên biến mảng.

- **value1**,...**valuen** là các giá trị của mảng mà các bạn muốn tạo.

Ví dụ

var arr = [1, 2, 4, 5, 9, 6];

Cách 2: Khai báo bằng new Array()

Cú pháp

var arr = new Array(value1, value2, ..., valuen);

Trong đó

- **arr** là tên biến mảng.

- **value1**,...**valuen** là các giá trị của mảng mà các bạn muốn tạo.

Ví dụ

var arr = new Array(1, 2, 4, 5, 9, 6);

2.5.2. Truy xuất mảng

Sau khi đã tạo được mảng rồi thì chúng ta cần phải truy xuất thông tin của mảng. Để có thể lấy ra giá trị của một thành phần trong mảng thì chúng ta sử dụng cú pháp sau:

arr[index];

Trong đó

arr là tên biến mảng, **index** là vị trí của mảng(bắt đầu từ 0).

Ví dụ

```
var hv=["Hùng","Minh","Thuỷ","Sơn"];
document.write(hv[2]);
```

3. Hàm trong Javascript

3.1. Hàm là gì?

Hàm là một hoặc nhiều đoạn mã được viết ra để thực thi một hoặc nhiều hành động khi gọi nó, hàm có khả năng gọi để thực thi các đoạn code viết bên trong nó nhiều lần, nhiều nơi. Mỗi hàm (funtion) thực hiện 1 chức năng nào đó của chương trình

3.2. Cấu trúc của hàm

Hàm trong javascript khai báo cũng giống như trong java, là bắt đầu phải có từ khóa ***function*** với các dạng như sau:

3.2.1. Hàm cơ bản

Đây là dạng cơ bản nhất của hàm trong javascript, có cú pháp như sau:

```
function funName() {
  //code
}
```

Trong đó: funName là tên hàm

Ví dụ: Tạo hàm lấy tên website

```
function getWebsite() {
   document.write('http://hoctot.cfi-elearning.edu.vn');
}
```

3.2.2. Hàm có tham số truyền vào

Cú pháp:

```
function funName(param_1, ..., pram_n) {
  //code
}
```

Trong đó:

- funName là tên của hàm các bạn muốn đặt.
- param_1,...,pram_n là các tham số mà các bạn muốn truyền vào hàm(không giới hạn số lượng)

Ví dụ: Tạo hàm tính tổng của 2 số bất kì

```
function getSum(a, b) {
  document.write('Tổng: ' + (a + b));
}
```

3.2.3. Hàm có tham số mặc định

Đây thực ra là dạng hàm có truyền tham số và đồng thời xét luôn giá trị mặc định cho các tham số đó

Cú pháp:

```
function funName(param_1 = value_1, ..., pram_n = value_2) {
  //code
}
```

Trong đó:

- funName là tên của hàm các bạn muốn đặt.
- param_1,...,pram_n là các tham số mà các bạn muốn truyền vào hàm(không giới hạn số lượng).
- value_1,...,value_n là các giá trị tương ứng với các pram.

Ví dụ:

```
function getSum(a = 1, b = 2) {
  document.write('Tổng: ' + (a + b));
}
```

3.2.4. Hàm có giá trị trả về và không

Trong javascript có hai loại hàm,đó là hàm có giá trị trả về và hàm không có giá trị trả về.

- Đối với hàm có giá trị trả về thì phải sử dụng từ khóa **return**

- Và ngược lại đối với hàm không có giá trị trả về thì không có từ khóa **return**.

Ví dụ:

```
function getSum(a = 1, b = 2) {
    return a + b;
}
```

3.3. Gọi hàm

Sau khi đã tạo được hàm muốn dùng nó chúng ta tiến hành thủ tục gọi hàm.

3.3.1. Hàm cơ bản

Để gọi loại hàm này sử dụng cú pháp:

funName();

Trong đó: **funName** là tên của hàm muốn gọi.

Ví dụ: Gọi hàm getWebsite ở trên.

```
getWebsite();
```

3.3.2. Hàm có tham số truyền vào

Để gọi dạng hàm này thì chúng ta cũng dùng cú pháp như hàm cơ bản, nhưng đồng thời truyền thêm các **param** vào theo cú pháp:

funName(param_1, ..., param_n);

Ví dụ: Ta sẽ gọi hàm getSum ở trên.

```
getSum(5, 6);
```

3.3.3. Hàm có tham số mặc định

Để gọi dạng hàm này chúng ta có thể sử dụng cách gọi hàm cơ bản nếu không muốn truyền tham số, và cách gọi hàm có tham số truyền vào nếu muốn truyền tham số cho hàm.

Ví dụ: Ở đây chúng ta có đoạn code sử dụng cả 2 cách

```
function getSum(a = 1,b = 2){
        document.write('Tổng: ' + (a + b));
    }
    getSum();
    //tạo vạch ngăn dễ nhìn
    document.write('<br>_____<br>');
    getSum(4,5);
```

3.3.4. Hàm có giá trị trả về và không

Ví dụ:

```
function getSum(a = 1,b = 2){
        return a + b;
    }
    var kq1 = getSum();
    document.write(kq1);
    //tạo khoảng ngăn cách
```

```
document.write('<br>_____________________________________<br>');
var kq2 = getSum(4,5);
document.write(kq2);
```

3.4. Các ràng buộc của tên hàm

Javascript cũng giống như các ngôn ngữ khác nó cũng có các ràng buộc về tên hàm sau đây:

- Tên hàm phải được bắt đầu bằng chữ cái (a-z,A-Z) hoặc ký tự _
- Tên hàm không được bắt đầu bằng số, các ký tự khác ký tự _

4. Sự kiện trong Javascript

4.1. DOM elements trong Javascript

DOM là viết tắt của chữ Document Object Model (Đối tượng tài liệu). Khi mà chúng ta làm việc với DOM thì chúng ta có thể tác động đến các thẻ HTML và các thành phần của nó. Trong bài này chúng ta sẽ chỉ tìm hiểu cách truy xuất đến các thẻ HTML thông qua class,id và tag.

4.1.1. Tìm thẻ HTML thông qua ID

Để tìm kiếm và truy xuất thông tin của một thẻ HTML theo ID của nó thì chúng ta sử dụng cú pháp:

document.getElementById('idName');

Trong đó: **idName** là id của thẻ HTML mà chúng ta cần truy xuất tới.

Ví dụ: Truy xuất và thông báo ra nội dung của thẻ HTML có id là **hs**.

C1. Viết chung js trong file html

```
<body>
  <div id="hello">Chào mừng các bạn đến với website học lập trình hoctot.cfi-elearning.edu.vn</div>
  <div id="hs">LẬP TRÌNH JAVASCRIPT</div>
  <script type="text/javascript">
        var element = document.getElementById('hs');
        var content = element.innerHTML;
        alert(content);
  </script>
  <script type="text/javascript"   src="3.js"></script>
</body>
```

C2. Viết phần javascript trong file 3.js

```
var element = document.getElementById('hs'); //Lấy phần tử có id=hs
var content = element.innerHTML; // gán nội dung phần tử id=hs cho biến content
alert(content); // Xuất hiện cảnh báo có nội dung là content
```

4.1.2. Tìm thẻ HTML thông qua class

Để tìm thẻ html thông qua class của nó sử dụng cú pháp như sau:

document.getElementsByClassName('className');

Trong đó: **className** là tên class của thẻ HTML muốn tìm

Ví dụ: Tìm và lấy ra nội dung của thẻ HTML có class name là hello.

```
<body>
  <div class="hello">Chào mừng các bạn đến với website học lập trình online</div>
  <div class="hello">Lấy nội dung CLASS</div>
```

```
  <script type="text/javascript">
        var element = document.getElementsByClassName('hello');
        //Lấy ra nội dung của thẻ đầu tiên
        var content = element[0].innerHTML;
        alert(content);
  </script>
<script type="text/javascript"   src="3.js"></script>
</body>
```

4.1.3. Tìm thẻ HTML thông qua tagName

Để tìm kiếm thẻ HTML có tagName theo ý bạn trong javascript thì chúng ta sử dụng cú pháp:

document.getElementsByTagName('tagName');

Trong đó: **tagName** là tag mà các bạn muốn tìm kiếm và truy xuất nó.

Ví dụ:

```
<script type="text/javascript">
    var element = document.getElementsByTagName('div');
    //Lấy ra nội dung của thẻ đầu tiên
    var content = element[0].innerHTML;
    alert(content);
</script>
```

Chú ý

- Đối với tìm thẻ HTML theo class và tag name thì nó sẽ trả về mảng đối tượng với số lượng phần tử bằng với số lượng các thẻ html thoả mãn điều kiện.

Ví dụ:

```
<body>
  <div id="hello">Chào mừng các bạn đến với website học lập trình online </div>
  <div class="bomb">JAVASCRIPT</div>
  <p class="bomb">Bacsic to advance</p>
  <script type="text/javascript">
   //chọn tất cả các phần tử có class là toidicode
   var element = document.getElementsByClassName('bomb');
   //lấy ra số lượng phần tử tìm đc
   var length = element.length;
   alert('Số phần tử có class là bomb = '+length);
  </script>
</body>
```

4.1.4. Tìm kiếm thành phần con trong thành phần

Có 2 cách

C1: Sử dụng lồng nhau các getElement

Ví dụ:

```
<body>
  <div id="hello">Chào mừng các bạn đến với website học lập trình online</div>
```

```
<div class="cha">ELEARNING
  <p class="con1">Con trai 1</p>
  <b class="con1">Con trai 1 đậm</b>
  <p class="con2">Con gái 2</p>
  <i class="con2">Con gái 2 nghiêng</i>
</div>

<script type="text/javascript">
   var ptcha = document.getElementsByClassName('cha');
   var ptcon1 = ptcha[0].getElementsByClassName('con1');
   var ptcon2 = ptcha[0].getElementsByClassName('con2');
   alert(ptcon2[1].innerText);
</script>
</body>
```

C2: Sử dụng querySelectorAll().

Với cú pháp:

document.querySelectorAll('select')

Trong đó: **select** các bạn sử dụng cú pháp như css.

Ví dụ:

```
<body>
  <div id="hello">Chào mừng các bạn đến với website học lập trình online</div>
<div class="cha">ELEARNING
  <p class="con1">Con trai 1</p>
  <b class="con1">Con trai 1 đậm</b>
  <p class="con2">Con gái 2</p>
  <i class="con2">Con gái 2 nghiêng</i>
</div>

<script type="text/javascript">
   var ptcha = document.querySelectorAll('div#hello');
   var ptcon1 = ptcha[0].innerText;
   var ptcon2 = ptcha[0].innerText;
   alert(ptcon1);
</script>
</body>
```

Khi chạy đoạn code này sẽ xuất hiện thông báo " Chào mừng các bạn đến với website học lập trình online" (div#hello là cái div có id = hello)

Tip: *Lưu ý* ***select*** *chúng ta sử dụng cú pháp như css. Có nghĩa bạn hãy thay thế div#hello lần lượt các class:* div.cha .con1 *hoặc* div.cha .con2; *và thay thế trong hàm alert(ptcon1) là alert(ptcon2), để thấy sự thay đổi*

4.2. DOM HTML trong javascript

Sau khi đã tìm kiếm được các thẻ HTML trong javascript thì giờ chúng ta sẽ cùng đi tìm hiểu các hàm tác động vào nó. Cụ thể phần này chúng ta sẽ tìm hiểu các hàm lấy ra giá trị của các thẻ HTML đó.

4.2.1. Lấy nội dung trong thẻ HTMl và thay đổi nó.

a. Lấy nội dung

-Để lấy nội dung bên trong thẻ HTMl chúng ta sử dụng cú pháp:

element.innerHTML

Trong đó: **element** là các đối tượng HTML (được gọi bằng các phương thức getElement...)

Ví dụ:

```
//lấy nội dung của thẻ có id = hello
var content = document.getElementById('hello').innerHTML;
//in ra nội dung của thẻ đó.
alert(content);
```

Hoặc cũng có thể viết như sau:

```
<script type="text/javascript">
    var phantu = document.getElementById('hello');   //lấy phần tử của thẻ có id = hello
    var content =phantu.innerHTML;  //lấy nội dung của thẻ có id = hello
    alert(content); //in ra nội dung của thẻ đó.
  </script>
```

b. Thay đổi nội dung

Để thay đổi nội dung cho thẻ HTML đó thì các bạn chỉ sử dụng cú pháp:

element.innerHTML = 'Giá trị mới';

Trong đó: **element** là các đối tượng HTML (được gọi bằng các phương thức getElement...)

Ví dụ:

```
<body>
  <div id="hello">Chào mừng các bạn đến với website học lập trình online </div>
   <div class="bomb">hoctot.cfi-elearning.edu.vn
   </div>

   <script type="text/javascript">
          var noidungold=document.getElementById('hello').innerHTML;
          alert(noidungold);
      //thay đổi nội dung của thẻ có id = hello
      document.getElementById('hello').innerHTML = 'Học javascript cơ bản';
      var noidungnew=document.getElementById('hello').innerHTML;
      alert(noidungnew);
   </script>
</body>
```

c. Lấy và thay đổi nội dung dưới dạng văn bản.

- Lấy nội dung

Nếu muốn lấy nội dung text, hay thay đổi nội dung dưới dạng dữ liệu thô thì chỉ cần thay thuộc tính innerHTML thành innerText.

Ví dụ:

```
<body>
  <div id="ecfi">  Chào mừng các bạn đến với <a href="http://hoctot.cfi-elearning.edu.vn/">website học lập trình online cfi</a> </div>
   <div class="cfi"><a href="http://cfi.edu.vn/">website trường Cao đẳng LTTP</a>
   </div>
 <script type="text/javascript">
      var content_ecfi = document.getElementById('ecfi').innerText; //lấy nội dung của thẻ có id = ecfi
     alert(content_ecfi); //in ra nội dung của thẻ có id = ecfi
     // ------------------------------------------
     var element = document.getElementsByClassName('cfi'); // Tìm thẻ có class=cfi
     var content_cfi=element[0].innerText; //Lấy text của phần tử đầu tiên có class=cfi
     alert(content_cfi); //in ra nội dung của thẻ có class = cfi
</script>
  </body>
```

Lưu ý: Trong ví dụ các thẻ div đều chứa bên trong các thẻ a liên kết, nên muốn lấy chữ (Text) thì chúng ta phải dùng InnerText, chứ không thể dùng InnerHTML.

* Nếu là id (duy nhất) thì chúng ta trực tiếp lấy nội dung text (bằng lệnh var content_ecfi = document.getElementById('ecfi').innerText)

* Nếu là class (không duy nhất vì sẽ có nhiều phần tử cùng class) thì chúng ta qua 2 bước:

B1: Tìm phần tử (var element = document.getElementsByClassName('cfi');)

B2: Lấy text của 1 phần tử (var content_cfi=element[0].innerText;)

- Thay đổi nội dung:

```
<body>
  <div id="ecfi">  Chào mừng các bạn đến với <a href="http://hoctot.cfi-elearning.edu.vn/">website học lập trình online cfi</a> </div>
   <div class="cfi"><a href="http://cfi.edu.vn/">website trường Cao đẳng LTTP</a>
   </div>

   <script type="text/javascript">
      var content_ecfi = document.getElementById('ecfi').innerText='<b>Học javascript cơ bản</b>';
      alert(content_ecfi);
      var element = document.getElementsByClassName('cfi');
      var content_cfi=element[0].innerText='<i> Lop 19I </>';
      alert(content_cfi);
   </script>
</body>
```

4.2.2. Tạo và lấy nội dung của các attribute trong thẻ HTML.

-Để tạo ra các attribute có các thẻ HTML chúng ta sử dụng cú pháp:

```
element.setAttribute('attrName','attrValue');
```

- Muốn lấy giá trị 1 thuộc tính dùng cú pháp:

```
element.getAttribute('attrName');
```

Trong đó:

- **element** là các đối tượng HTML (được gọi bằng các phương thức getElement...).
- **attrName** là tên của attribute mà các bạn muốn tạo ra.
- **attrValue** là giá tri của attribute các bạn muốn set.

Ví dụ:

```
<body>
        <div id="hello">  Chào mừng các bạn đến với
         <a href="http://.com">website học lập trình online</a>
        </div>
        <div class="ecfi">ELEARNING     </div>

       <script type="text/javascript">

          var am = document.getElementById('hello'); //1
          am.setAttribute('data-id','5'); //2
          //-----------------------------------
          var ele =document.getElementById('hello').innerHTML;
          console.log(ele);
         //-----------------------------------
          var atr=document.getElementById('hello').getAttribute('data-id'); //3
          console.log(atr); //4
       </script>
</body>
```

Giải thích code:

- Lệnh 3 khai báo biến am để lưu phần tử có d=hello, lệnh 4 thiết lập thêm thuộc tính data-id=5; 2 lệnh này có thể thay bằng 1 lệnh: document.getElementById('hello'). setAttribute('data-id','5');

- Lệnh 3 khai báo biến atr để lưu giá trị thuộc tính data-id

4.3. DOM CSS trong javascript

Dựa vào hàm **setAttribute** trong phần trước thì chúng ta hoàn toàn có thể thêm css cho thẻ HTML, nhưng trong javascript còn hỗ trợ chúng ta một cách CSS chuyên nghiệp hơn nữa.

Cú pháp

1. Để CSS cho các thẻ HTMl bằng javascript dùng cú pháp:

```
element.style.propertyName = value;
```

Trong đó:

- **element** là các đối tượng HTML (được gọi bằng các phương thức getElement...).
- **propertyName** là tên thuộc tính CSS các bạn muốn xét.
- **value** là giá trị của thuộc tính đó.

2. Và để lấy ra giá trị của thuộc tính css sử dụng cú pháp:

```
element.style.propertyName;
```

Ví dụ: Thiết lập color=red cho thẻ HTML có id=series; color=#326CC4 cho class=list

```
<div>Các series học lập trình tại hoctot.elearning.edu.vn</div>
  <ul>
    <li class="list">Học HTML</li>
    <li id="series">Học CSS</li>
    <li class="list">Học javascript</li>
    <li class="series">Học jquery</li>
  </ul>

  <script type="text/javascript">
    //tìm đến thẻ HTML có id = series và css
    document.getElementById('series').style.color = 'red';
      //tìm đến các thẻ HTML có class = list và css
     var ele=document.getElementsByClassName('list');
    for (var i = 0; i < ele.length; i++) {
          ele[i].style.color='#326CC4';
    }
  </script>
```

Ví dụ: Lấy ra giá trị của thuộc tính css.

```
<body>
 <div>Các series học lập trình </div>
  <ul>
    <li class="list">Học HTML</li>
    <li id="series" style="background:#FED800 ">Học CSS</li>
    <li class="list">Học javascript</li>
    <li class="series">Học jquery</li>
  </ul>

  <script type="text/javascript">
    //tìm đến thẻ HTML có id = series và lấy thông số của thuộc tính background
    var value = document.getElementById('series').style.background;
    //hiển thị ra giá trị
    alert(value);
  </script>
</body>
```

Chú ý: *Đối với các thuộc tính css có dấu - ví dụ như: background-color, margin-top, padding-left thì chúng ta phải bỏ dấu - và chuyển lại như sau: backgroundColor, marginTop, paddingLeft*

4.4. DOM events trong javascript

4.4.1. Thêm sự kiện bằng HTML attribute.

- Các thẻ HTML có rất nhiều thuộc tính (attribute) tuy nhiên các thẻ lại có một số thuộc tính như: onClick, onMouseover,.... Dùng để bắt các sự kiện và thực thực thi hành động bằng javascript. Và giá trị của các thẻ này là code javascript.

Ví dụ:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>DOM Events in JavaScript</title>
</head>
<body>
  <div >Click vào chữ Học css để xem kết quả</div>
   <div id="series" onclick="alert('Bạn vừa click vào đấy')">Học CSS</div>
</body>
</html>
```

Giải thích: khi chạy trên trình duyệt, nếu bạn bấm vào chữ Học CSS *thì cái thông báo* 'Bạn vừa click vào đấy' *xuất hiện*

- Chúng ta vẫn có thể gọi hàm trong attribute

Ví dụ:

```
<body>
   <div >Di chuột vào chữ Học css để xem kết quả</div>
   <div id="series" onmouseover="show()">Học CSS</div>
   <script type="text/javascript">
      function show(){
         alert('Bạn vừa rê chuột vào')
      }
   </script>
</body>
```

4.4.2. Thêm sự kiện bằng Javascript

-Để thêm sự kiện bằng javascript chúng ta sử dụng cú pháp:

```
element.eventName = function ()
{
   //code
}
```

Trong đó:

- element là các đối tượng HTML (được gọi bằng các phương thức getElement...).
- eventName là tên các sự kiện như onclick,onmouseover,...

Ví dụ:

```
<body>
    <div >Di chuột vào chữ Học javascript để xem kết quả</div>
```

```
    <div id="jav" >Học javascript</div>
    <script type="text/javascript">
        //chọn thẻ có id = jav
        var element = document.getElementById('jav');
        //Thêm sự kiện
        element.onmouseover = function (){
            alert('Bạn vừa di chuột vào');
        };
    </script>
</body>
```

Ví dụ: Trong trường hợp mà đối tượng HTML các bạn chọn có nhiều hơn 1 thì các bạn cần phải chạy một vòng for để duyệt và add sự kiện cho nó.

```
    <body>
        <div >Di chuột vào chữ từng Li để xem kết quả</div>
    <ul>
        <li class="list">Li 1</li>
        <li class="list">Li 2</li>
        <li class="list">Li 3</li>
        <li class="list">Li 4</li>
        <li class="list">Li 5</li>
        <li class="list">Li 6</li>
        <li class="list">Li 7</li>
    </ul>
    <script type="text/javascript">
        //chọn thẻ có class = list
        var element = document.getElementsByClassName('list');
        //Thêm sự kiện
        for(i=0; i<element.length; i++) {
            element[i].onclick = function () {
                alert('Bạn vừa click vào thẻ ' + this.innerText);
            };
        }
    </script>
</body>
```

Tip: *This* *là gì? This trong javascript nó cũng giống như các ngôn ngữ khác, đều là đại diện cho đối tượng hiện tại mà nó đang được sử dụng.*

Ví dụ:

```
    <body>
        <div>Click vào các button để xem kết quả</div>
        <button onclick="showInfo(this)" > button1</button>
        <button onclick="showInfo(this)" > button2</button>
```

```
        <button onclick="showInfo(this)" > button3</button>
        <button onclick="showInfo(this)" > button4</button>
        <script type="text/javascript">
            function showInfo(e)
            {
                alert(e.innerText);
            }
        </script>
    </body>
```

4.4.3. Bảng danh sách sự kiện

Sự kiện	HTML5	Mô tả
onafterprint	√	Kích hoạt sau khi một tài liệu được in
onbeforeprint	√	Kích hoạt trước khi một tài liệu được in
onbeforeonload	√	Kích hoạt trước khi một tài liệu được tải
onerror	√	Kích hoạt khi một lỗi xảy ra
onhaschange	√	Kích hoạt khi một tài liệu đã thay đổi
onload		Kích hoạt khi một tải liệu được tải
onmessage	√	Kích hoạt khi một thông báo được kích hoạt (chạy)
onoffline	√	Kích hoạt khi một tài liệu ở ngoại tuyến (bị ngắt mạng)
ononline	√	Kích hoạt khi một tài liệu ở dạng trực tuyến
onpagehide	√	Kích hoạt khi một cửa sổ bị ẩn
onpageshow	√	Kích hoạt khi một cửa sổ trở lên được nhìn thấy (hiện lên)
onpopstate	√	Kích hoạt khi lịch sử của cửa sổ thay đổi
onredo	√	Kích hoạt khi một tài liệu thực hiện một redo
onresize	√	Kích hoạt khi một cửa sổ được thay đổi lại kích thước
onstorage	√	Kích hoạt khi một tài liệu được tải
onundo	√	Kích hoạt khi một tài liệu thực hiện một undo
onunload		Kích hoạt khi một người sử dụng rời khỏi tài liệu
onclick		Kích hoạt trên con chuột vừa nhấn vào phần tử
ondblclick		Kích hoạt trên con chuột vừa nhấn đúp vào phần tử
ondrag		Kích hoạt khi một phần tử được kéo

ondragend	√	Kích hoạt ở phần cuối của thao tác kéo
ondragenter	√	Kích hoạt khi một phần tử đã được kéo tới một mục tiêu được thả xuống hợp lệ
ondragleave	√	Kích hoạt khi một phần tử rời khỏi một mục tiêu thả xuống hợp lệ
ondragover	√	Kích hoạt khi một phần tử đang được kéo qua một mục tiêu có thể thả xuống hợp lệ
ondragstart	√	Kích hoạt ở phần đầu của một hoạt động kéo
ondrop	√	Kích hoạt khi một phần tử được kéo đang được thả xuống
onmousedown		Kích hoạt khi một nút chuột (chuột trái hay phải) được nhấn
onmousemove		Kích hoạt khi con trỏ chuột di chuyển
onmouseout		Kích hoạt khi con trỏ chuột rời khỏi một phần tử
onmouseover		Kích hoạt khi con trỏ chuột di chuyển qua một phần tử
onmouseup		Kích hoạt khi một nút chuột được thả ra
onmousewheel	√	Kích hoạt khi sử dụng bánh xe chuột
onscroll	√	Kích hoạt khi một thanh cuốn của phần tử được cuốn
onkeyup		Xảy ra khi bạn gõ phím nhưng lúc bạn nhả phím ra sẽ được kích hoạt
onkeypress		Xảy ra khi bạn nhấn môt phím vào ô input
onblur		Xảy ra khi con trỏ chuột rời khỏi ô input
oncopy		Xảy ra khi bạn copy nội dung của thẻ
oncut		Xảy ra khi bạn cắt nội dung của thẻ
onpaste		Xảy ra khi bạn dán nội dung vào thẻ

4.4.4. Bài tập có hướng dẫn

Bài 1

Viết chương trình gồm một ô input và một thẻ div dùng để hiển thị nội dung (giá trị của ô input) khi người dùng gõ vào ô input.

Vì đề bài yêu cầu khi nhập dữ liệu vào ô input thì hiển thị nội dung bên trong thẻ DIV nên ta có thể sử dụng sự kiện onkeyup. Thứ hai nữa là chúng ta sẽ sử dụng các hàm DOM Element để truy xuất các đối tượng HTML.

Code

```
<body>
    <input type="text" id="message" value="" onkeyup="show_result()">
   <div id="result"></div>
```

```
<script type="text/javascript">
        // Hàm show kết quả
  function show_result()  {
    // Lấy hai thẻ HTML
    var t_input = document.getElementById("message");
    var t_div = document.getElementById("result");
   // Gán nội dung ô input vào thẻ div
     t_div.innerHTML = t_input.value; // Lưu ý: t_input.value; là lệnh lấy giá trị hiện tại thuộc tính value của phần tử t_input
  }
</script>
</body>
```

Giải thích: *Khi chúng ta nhập nội dung vào ô input thì do có sự kiện onkeyup (lúc nhả chuột ra nó thực hiện hàm show_result, mà hàm này có nhiệm vụ chuyển nội dung đã nhập vào thẻ div*

Bài 2

Viết chương trình khi người dùng copy nội dung của website thì thông báo là bạn đã copy thành công (sử dụng oncopy)

```
<body>
 <h3>Hãy copy dòng chữ dưới đây:</h3>
  <div oncopy="show_message()">Chào mừng các bạn đến với website hoctot.cfi-elearning.edu.vn</div>
  <script language="javascript">
   // Hàm show kết quả
   function show_message()
   {
    alert("Bạn đã copy thành công");
   }
  </script>
</body>
```

Bài 3

Viết chương trình tính tổng của hai số nhập vào (tính tự động). Bài này ta phải tạo 3 ô input và gán sự kiện onkeyup cho 2 ô input đầu tiên, trong sự kiện này sẽ thực hiện tính tổng của hai ô và in kết quả vào ô input thứ 3.

Code

```
<body>
  Nhập số a: <input type="text" id="a" value="" onkeyup="tinh()"> <br><br>
  Nhập số b: <input type="text" id="b" value="" onkeyup="tinh()"> <br><br>
  Kết quả là: <input type="text" id="result" value=""/>
 <script language="javascript">
   // Hàm tính kết quả
   function tinh()
   {
```

```
    // Lấy 3 ô input
    var a = document.getElementById("a");
    var b = document.getElementById("b");
    var result = document.getElementById("result");

    // Tính tổng hai ô đầu tiên
    var tong = parseInt(a.value)
+ parseInt(b.value);

    // Gán giá trị vào ô thứ ba
    // Phải kiểm tra tổng hai số này có bị lỗi hay không
    if (!isNaN(tong)){
      result.value = tong;
    }
  }
 </script>
</body>
```

Giải thích:

1. Phương thức parseInt(): sẽ phân tích một chuỗi và trả về một số nguyên nếu có thể, phương thức parseInt() sẽ chỉ trả về một số nguyên kể cả khi chuỗi được bắt đầu bằng một số thập phân, chính vì điểm khác nhau trên, nếu chuỗi được bắt đầu bằng một dấu chấm, phương thức sẽ trả về NaN.
   - Các khoảng trắng ở đầu và cuối chuỗi sẽ không ảnh hưởng đến kết quả.
   - Nếu kí tự đầu tiên của chuỗi không thể chuyển thành kiểu number, phương thức sẽ trả về NaN.

   Cú pháp được sử dụng:

   parseInt(string, radix)

Trong đó:

- string là chuỗi cần phân tích.
- radix là tham số không bắt buộc.
- Nếu radix không được truyền:
  - Nếu chuỗi bắt đầu bằng '0x', radix sẽ mang giá trị 16(hệ thập lục phân).
  - Nếu chuỗi bắt đầu bằng bất kì số nào khác, radix sẽ mang giá trị 10(hệ thập phân).
- Nếu được truyền vào, radix(từ 2 đến 36) radix sẽ đại diện cho hệ cơ số sẽ được sử dụng để phân tích chuỗi

2. Phương thức isNaN() sẽ kiểm tra điều kiện giá trị truyền vào có phải là một số không hợp lệ(Not-A-Numbet) hay không. Phương thức sẽ trả về True nếu tham số truyền vào là một giá trị NaN, ngược lại phương thức sẽ trả về False.

4.4.5. Luyện tập

Bài 1: Hãy viết chương trình khi click vào một button thì xuất hiện một thông báo với nội dung "Chào mừng bạn đến với hoctot.cfi-elearning.edu.vn".

Gợi ý: Tạo một input, sau đó thêm sự kiện khi onclick vào input đó thì sẽ alert lên câu thông báo như đề bài đưa ra.

Bài 2: Sử dụng kiến thức đã học hãy viết chương trình gồm 2 button và 1 thẻ div, khi click vào button thứ nhất thì thiết lập nội dung cho thẻ div là "Demo thay đổi nội dung thẻ div bằng Javascript tại CFI", khi click vào button thứ hai thì gán nội dung là "Nội dung của thẻ div đã bị thay đổi".

Gợi ý: Để thay đổi nội dung của một thẻ HTML thì ta sử dụng thuộc tinh innerHTML, nhưng trước khi sử dụng thuộc tính đó thì ta phải sử dụng DOM để lấy thẻ div cần thay đổi nội dung và gán sự kiện onclick vào 2 button.

Bài 3: Cho một đoạn mã HTML có sẵn như dưới đây, hãy viết một đoạn mã Javascript khi click vào button#btn1 thì đổi màu background cho thẻ div#content sang màu đỏ, khi click vào button#btn2 thì đổi background sang màu xanh.

```
<h2>Thay đổi CSS cho thẻ HTML</h2>
<div id="content" style="padding: 20px; background: #ddd; color: white; margin: 40px 0px;">
  CHÀO MỪNG BẠN ĐẾN VỚI CFI-ELEARNING
</div>
<input type="button" id="btn1" value="Đổi Backgroud màu đỏ"/>
<input type="button" id="btn2" value="Đổi Backgroud màu xanh"/>
```

4.5. Return TRUE/FALSE của Events trong Javascript

Trong một sự kiện sẽ có hai trạng thái là hành động đúng (TRUE) và hành động sai (FALSE) và để thể hiện hai trạng thái này thì ta sử dụng cú pháp return TRUE/FAlSE trong function của event. Đây là một vấn đề khá quan trọng đấy nếu bạn sử dụng không đúng thì bạn sẽ không biết tại sao mình code sai và không biết chỗ nào để sửa cả.

Có hai cách return thông dụng nhất như sau:

Cách 1: Return tại đoạn code event trong HTML luôn, ví dụ

```
<input type="text" onkeypress="return false" />
```

Cách 2: Tạo một hàm xử lý sự kiện, lúc này hàm này phải return và trong HTML bạn cũng phải return, ví dụ:

```
<script language="javascript">
function check()
{
   return false;
}
</script>
<input type="text" onkeypress="return check()" />
```

Chúng ta phải thực sự hiểu ý nghĩa của từng loại sự kiện trong javascript thì mới áp dụng đúng được. ví dụ mình cần viết chương trình không cho người dùng nhập bất kỳ chữ gì vào một ô input thì nếu sử dụng sự kiện onkeyup sẽ sai, lý do là sự kiện này sảy ra khi nhã phím nên bạn nhập rồi nó mới kích hoạt. Thay vì vậy thì bạn sử dụng sự kiện onkeypress sẽ thành công.

TÀI LIỆU THAM KHẢO

[1]. VN-Guide, *Thiết kế trang web với HTML*, NXB Thống kê HN, 2004;
[2]. VN Guide, *Internet toàn tập*, NXB Khoa học kỹ thuật, 1995;
[3]. SCC Technology, *Thiết kế website với Macromedia Dreamweaver*, 2004;
[4]. Nguyễn Trường Sinh, *Thiết kế website với FrontPage*, NXB Thống kê, 2006;
[5]. https://getbootstrap.com/docs/4.0/getting-started/download/

MỤC LỤC