1

Thiết kế WEB

Nguyễn Đình Hoàng Sơn

BM Hệ thống thông tin – Khoa Công nghệ thông tin

Trường Đại học Nha Trang

Email: sonndh@ntu.edu.vn - DĐ: 083 8705124

0

2

Nội dung

- Các khái niệm
- HTML
- CSS
- Javascript
- Các ứng dụng (Jquery + Bootstrap)

1

3

Tài liệu tham khảo

- Thiết kế trang web – Nguyễn Đình Hoàng Sơn
- https://www.w3schools.com/
- https://viblo.asia/get-started
- https://getbootstrap.com/

4

Internet

❖ **Internet :** liên mạng máy tính toàn cầu được hình thành từ các mạng nhỏ hơn, liên kết máy tính trên thế giới thông qua cơ sở hạ tầng viễn thông

5

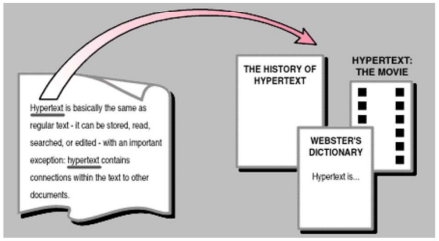
World Wide Web

❖**World Wide Web :** dịch vụ cho phép người dùng khai thác thông tin trên Internet thông qua một số công cụ hoặc là chương trình hoạt động dưới các giao thức mạng.

6

Các thành phần của WEB

❖**Hypertext :** là một loại văn bản thông thường nhưng lại có tham chiếu (**hyperlink** – liên kết) tới các văn bản khác.



7

Các thành phần của WEB

❖**Webpage :** là hypertext có thể được truy cập bởi phần mềm trình duyệt (web browser) và hiển thị trên màn hình máy tính hoặc thiết bị điện tử (*ví dụ*: điện thoại di động)

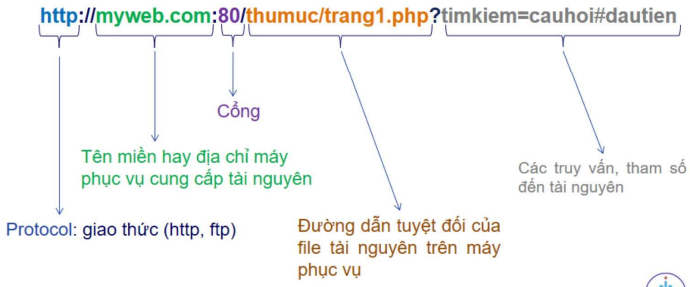
➢ Webpage được truy cập trên mạng Internet thông qua địa chỉ gọi là Uniform Resource Locator (URL)

❖**Website :** là tập hợp các tài liệu web (webpage, file định dạng CSS, các script, hình ảnh, âm thanh…) có liên quan với nhau, được đặt tại ít nhất một máy tính (*web server*)

8

Các thành phần của WEB

❖ **Uniform Resource Locator :** Là chuỗi định vị tài nguyên trên Internet



9

Các thành phần của WEB

➢ **Cổng dịch vụ (Service port)**

– Một Server có thể cung cấp nhiều dịch vụ → cần sử dụng cổng để xác định dịch vụ cung cấp.

– Mỗi dịch vụ thường chiếm những cổng mặc định.

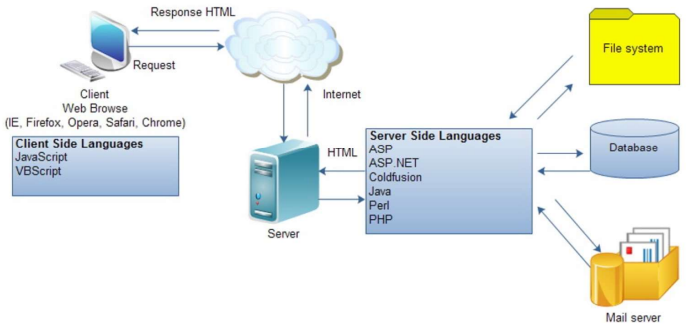
80	:	Web service
21	:	FTP
25	:	SMTP
110	:	POP3
1521	:	CSDL Oracle

– Có thể sử dụng PORT MAPPING để chuyển đổi số port mặc định của một dịch vụ nào đó đến 1 số khác.

Ví dụ : http://www.abc.com:8080/

10

Mô hình Client – Server



11

Quy trình thiết kế website

1. Đặt kế hoạch

- ➢ Mục tiêu
- ➢ Tính khả thi
  - ✓ *Thời gian*
  - ✓ *Nhân sự*
  - ✓ *Tài chính*

12

Quy trình thiết kế website

2. Phân tích : hiểu rõ yêu cầu của khách hàng, xác định điều kiện cần thiết của hệ thống.

- Thu tập thông tin để xác định vấn đề cần giải quyết.
- Xác định yêu cầu hệ thống
- Nội dung, hình ảnh?
- Thứ tự và mối liên quan giữa các nội dung

…

13

Quy trình thiết kế website

3. Thiết kế

- Sơ đồ cấu trúc website
- Giao diện
- CSDL
- Nội dung từng trang
- Liên kết giữa các trang

…

14

Quy trình thiết kế website

4. Xây dựng và cài đặt website

- Viết code
- Chọn tên miền và host
- Cài đặt website vào hệ thống
- Kiểm thử

15

Một số điều cần chú ý trong lập trình WEB

5. Các hoạt động hỗ trợ

- Bảo trì: sửa lỗi, cập nhật
- Mở rộng: thay đổi chức năng, nâng cấp hệ thống
- Hướng dẫn người dùng

16

Lịch sử phát triển của công nghệ Web

- 1991 HTML (Tim Berners-Lee tạo 20 thẻ HTML đầu tiên)
- 1994 HTML 2
- 1996 CSS 1 + JavaScript
- 1997 HTML 4
- 1998 CSS 2
- 2000 XHTML 1
- 2002 Tableless Web Design
- 2005 AJAX
- 2014 HTML 5

17

HTML5

Offline / Storage
Realtime / Communication
File / Hardware Access
Semantics & Markup
Graphics / Multimedia
CSS3
Nuts & Bolts

Lưu trữ
Kết nối
Thiết bị truy cập
Ngữ nghĩa
Đồ họa 3D,hiệu ứng
Css3
Đa phương tiện
Hiệu năng

18

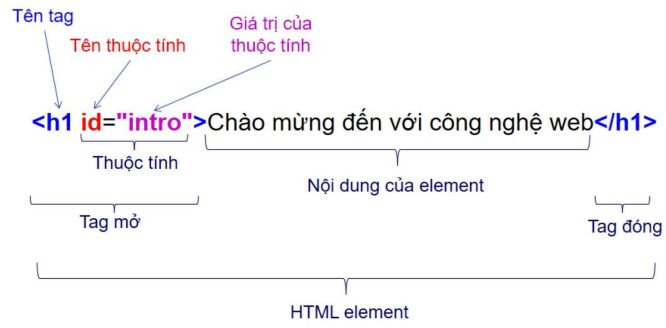
HTML

❖**HTML (HyperText Markup Language) :** ngôn ngữ được dùng để mô tả cấu trúc và nội dung của văn bản siêu liên kết (hypertext)

- HTML không phải là ngôn ngữ lập trình
- HTML không phải là ngôn ngữ "định dạng"
- HTML sử dụng những mã "đánh dấu" – ***phần tử (element)*** – để mô tả các thành phần của văn bản siêu liên kết.

19

HTML Element



20

Cấu trúc tài liệu HTML

```
<!DOCTYPE html>
<html lang="vi">
  <head>
    <meta charset="UTF-8" />
    <title>Tiêu đề trang web</title>
    <link rel="stylesheet" href="fileCSS_1.css" />
    <link rel="stylesheet" href="fileCSS_2.css" />
  </head>
  <body>
    Nội dung trang web.....
  </body>
</html>
```

❖ **Cho phép khai báo nhiều file CSS bằng thẻ <link>**

21

Cấu trúc tài liệu HTML

☞ **Một số chú ý :**

- Một số tag đóng không bắt buộc phải có.
- Một số tag không theo cặp – không có tag đóng

  Ví dụ: **
**
- Các element được phân loại: ***element khối*** (block-level element) và ***element dòng*** (inline element)
  - *Block-level element* : nội dung của element khi hiển thị trên trình duyệt sẽ tự xuống dòng để phân cách với các element khác (*ví dụ*: <h1>…</h1> hoặc …)
  - *Inline element* : nội dung của element sẽ tiếp tục ở trên dòng hiển thị của browser (*ví dụ*: …)

22

Một số ký tự đặc biệt trong HTML

Kết quả	Loại	Thể hiện trong văn bản HTML	Số
	một khoảng trắng		
<	nhỏ hơn	&lt;	&#60;
>	lớn hơn	&gt;	&#62;
&	dấu và	&amp;	&#38;
"	nháy kép	&quot;	&#34;
'	nháy đơn	&apos;	&#39;

23

Một số ký tự đặc biệt trong HTML

Kết quả	Loại	Thể hiện trong văn bản HTML	Số
¢	cent	&cent;	&#162;
£	pound	&pound;	&#163;
¥	yen	&yen;	&#165;
§	section	&sect;	&#167;
©	copyright	&copy;	&#169;
®	registered trademark	&reg;	&#174;
×	nhân	&times;	&#215;
÷	chia	&divide;	&#247;

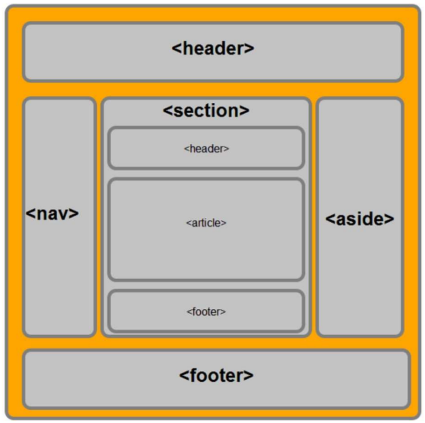
24

Các thẻ (tag) cơ bản trong HTML

Tag	Mô Tả
<html>...</html>	Xác định một văn bản dạng HTML
<body>...</body>	Xác định phần thân của tài liệu
...	Xác định một vùng
... ...	Xác định header từ 1 đến 6
...	Xác định một đoạn văn
 	Chèn một dòng trắng
<hr />	Chèn một đường thẳng

25

Thẻ bố cục (HTML5)



26

Thẻ bố cục (HTML5)

- **<header>** : định nghĩa phần đầu của trang web
- **<nav>** : thanh điều hướng (menu)
- **<section>** : định nghĩa 1 phần website
- **<article>** : định nghĩa nội dung bài viết độc lập
- **<aside>** : định nghĩa phần bên cạnh nội dung
- **<footer>** : định nghĩa đáy (chân) trang
- **<details>** : định nghĩa thông tin chi tiết
- **<summary>** : định nghĩa tiêu đề tóm tắt của 1 phần tử details

27

Thẻ liên kết

```
<a href="link liên kết - URL">Text_liên_kết</a>
```

Ví dụ: Tìm kiếm bằng Google

Link tới trang sample

```
<a href="mailto:abc@yahoo.com">
        Link tới email abc@yahoo.com
</a>
```

28

Thẻ liên kết – liên kết trong

```
<a name="vị_trí_đánh_dấu">Text_liên_kết</a>

<a href="#vị_trí_đánh_dấu">Text_liên_kết</a>
```

```
<html>
<body>
    <p><a href="#C2">Tới chapter 2</a></p>

    <h2>Chapter 1</h2>
    <p>Chapter 1 là .....</p>

    <h2><a name="C2">Chapter 2</a></h2>
    <p>Chapter 2 là.....</p>
</body>
</html>
```

29

Tạo bảng

Tag	Mô Tả
<table>…</table>	Vẽ bảng
<th>…</th>	Hàng đầu của bảng
<tr>…</tr>	Hàng trong bảng
<td>…</td>	Ô trong hàng
<caption>…</caption>	Nhãn của bảng
<colgroup>…</colgroup>	Nhóm các cột
<col>…</col>	Định các thuộc tính của cột
<thead>…</thead>	Hàng Đầu bảng
<tbody>…</tbody>	Thân của bảng
<tfoot>…</tfoot>	Hàng cuối bảng

30

Tạo bảng

Ví dụ:

```
<html>
<body>
    <h4>Nhóm 2 ô hàng ngang thành 1 ô</h4>

    <table border="1">
        <tr>
            <td>Name</td>
            <td colspan="2">Telephone</td>
        </tr>
        <tr>
            <td>Bill Gates</td>
            <td>555 77 854</td>
            <td>555 77 855</td>
        </tr>
    </table>
</body>
</html>
```

Nhóm 2 ô hàng ngang thành 1 ô

Name	Telephone	
Bill Gates	555 77 854	555 77 855

31

Tạo bảng

Ví dụ:

```
<html>
<body>
    <h4>Nhóm 2 ô hàng dọc thành 1 ô</h4>

    <table border="1">
        <tr>
            <td>First Name:</td>
            <td>Bill Gates</td>
        </tr>
        <tr>
            <td rowspan="2">Telephone:</td>
            <td>555 77 854</td>
        </tr>
        <tr>
            <td>555 77 855</td>
        </tr>
    </table>
</body>
</html>
```

Nhóm 2 ô hàng dọc thành 1 ô

First Name:	Bill Gates
Telephone:	555 77 854
	555 77 855

32

Danh sách

Tag	Mô Tả
...	Danh sách có sắp xếp
...	Danh sách không sắp xếp
...	1 phần tử trong danh sách
<dl>...</dl> <dt>...</dt> <dd>...</dd>	Kiểu danh sách khác

Ví dụ:

```
<ul><strong>Thực đơn điểm tâm</strong>
    <li type="circle">Bò beefsteak</li>
    <li type="disc">Salad trộn</li>
    <li type="square">Nước trái cây ép</li>
</ul>
```

Thực đơn điểm tâm
- Bò beefsteak
- Salad trộn
- Nước trái cây ép

33

Danh sách

➢ ***Định dạng danh sách***

- *Định dạng cho từng mục*
  - ✓ <li **type="A"**> kí tự chữ hoa: A, B,…
  - ✓ <li **type="a"**> kí tự chữ thường: a, b,…
  - ✓ <li **type="I"**> kí tự LaMã: I, II…
  - ✓ <li **type="i"**> kí tự thường: i, ii,…
  - ✓ <ol **start="n"**> bắt đầu với thứ tự n
- *Định dạng cho toàn bộ danh sách*
  - ✓ <ol **type="giá_trị"**>…

34

Chèn ảnh

Tag	Mô Tả
<img>	Hình ảnh
<map>	Định nghĩa map (sơ đồ trên 1 hình)
<area>	Định area, 1 vùng nhỏ tương ứng trên sơ đồ

Ví dụ:

```
<img src="planets.gif" width="145" height="126" alt="Planets" usemap="#planetmap" />

<map name="planetmap">
    <area shape="rect" coords="0,0,82,126" href="sun.htm" alt="Sun" />
    <area shape="circle" coords="90,58,3" href="mercur.htm" alt="Mercury" />
    <area shape="circle" coords="124,58,8" href="venus.htm" alt="Venus" />
</map>
```

35

Multimedia

<video> : nhúng nội dung video

<audio> : nhúng nội dung âm thanh

<source>: khai báo (họ) định dạng multimedia

```
<video autoplay controls loop poster="img/video.jpg">
  <source src="multimedia/movie.mp4" type="video/mp4">
  <source src="multimedia/movie.ogg" type="video/ogg">
     Your browser does not support the video tag.
</video>
```

36

Đồ họa

<svg> : tạo đối tượng đồ họa

<canvas> : tạo vùng vẽ đồ họa tự do

```
<svg width="100" height="100">
  <circle cx="50" cy="50" r="40"
   stroke="green" stroke-width="4" fill="yellow"/>
</svg>
```

37

Đồ họa

```
<canvas id="myCanvas"></canvas>
<script>
  var c = document.getElementById("myCanvas");
  var ctx = c.getContext("2d");
  ctx.beginPath();
  ctx.arc(50, 50, 40, 0, 2*Math.PI);
  ctx.stroke();
</script>
```

38

Biểu mẫu - form

Tag	Mô Tả
<form>	Kiểu form để nhập thông tin
<input>	Một ô nhập liệu
<textarea>	Vùng nhập liệu có nhiều hàng
<label>	Nhãn
<fieldset>	Nhóm các vùng nhập với nhau
<legend>	Nhãn của 1 fieldset
<select>	Danh sách chọn
<option>	1 phần tử trong danh sách chọn
<optgroup>	Nhóm các phần tử trong danh sách chọn
<button>	Nút bấm

39

Biểu mẫu - form

Ví dụ:

```
<form action="form_action.php" method="get">
<fieldset>
    <legend>Member Infomation</legend>
    <label for="name">Full Name:</label>
        <input type="text" name="name" id="fname" size="30" />
    <br />
    <label for="pwd">Password:</label>
        <input type="password" name="pwd" id="pw" size="30" />
    <br />
    <label for="addr">Address:</label>
        <textarea name="addr" id="feedback" rows="10" cols="30" >
        </textarea> <br/>
    <input type="submit" value="Submit" id="submit"/>
    <input type="reset" value="Reset" name="reset"/>
</fieldset>
</form>
```

40

Biểu mẫu - form

➢ ***Các loại thẻ nhập liệu <input>***

- text
- password
- radio
- checkbox
- hidden
- submit
- reset
- button
- file
- image

41

Biểu mẫu - form

Ví dụ:

Male
Female

```
<form>
    <input type="radio" name="sex" value="male" checked="checked"/> Male
    <br>
    <input type="radio" name="sex" value="female"/> Female
</form>
```

Ví dụ:

I have a bike
I have a car

```
<form>
    <input type="checkbox" name="bike" checked="checked"/>I have a bike
    <br>
    <input type="checkbox" name="car"/>I have a car
</form>
```

42

Biểu mẫu - form

Ví dụ:

```
<select>
    <option value="1">Toyota</option>
    <option value="2" selected="selected">Honda</option>
    <option value="3">Mercedes</option>
    <option value="4">Ford</option>
</select>
```

Honda
Toyota
Honda
Mercedes
Ford

Ví dụ:

```
<select>
    <optgroup label="Swedish Cars">
        <option value="volvo">Volvo</option>
        <option value="saab">Saab</option>
    </optgroup>
    <optgroup label="German Cars">
        <option value="mercedes">Mercedes</option>
        <option value="audi">Audi</option>
    </optgroup>
</select>
```

Volvo
Swedish Cars
Volvo
Saab
German Cars
Mercedes
Audi

43

Biểu mẫu - form

✓ *Nếu thẻ <select> có thêm thuộc tính multiple thì sẽ có list box*

```
<select multiple="multiple">
    <option value="1">Toyota</option>
    <option value="2">Honda</option>
    <option value="3">Mercedes</option>
    <option value="4">Ford</option>
</select>
```

Toyota
Honda
Mercedes
Ford

44

HTML5 – Form

❖ Các thuộc tính kiểu nhập liệu mới: <input type="…" />

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

❖ Các thuộc tính nhập liệu mới

autocomplete	autofocus	form
height, width	list	min, max
multiple	pattern	placeholder
required	step	

❖ Các thuộc tính form: autocomplete novalidate

45

HTML5 – Form

<output> : xuất kết quả tính toán trong form

```
<form oninput="x.value = parseInt(a.value)
                        + parseInt(b.value)">
  0<input type="range" id="a" min="0" max="100" />100
  +<input type="number" id="b" />
  =<output name="x" for="a b"></output>
</form>
```

46

và

- Thẻ dùng để định nghĩa nội dung trong dòng (in-line) còn thẻ dùng để định nghĩa nội dung mức khối (block-level)

Ví dụ:

```
<div>
    <p>Cộng Hòa Xã Hội Chủ Nghĩa Việt Nam</p>
    <p>Độc lập - <span>Tự do</span> - Hạnh phúc</p>
</div>
```

- Thẻ và thường được dùng như công cụ đánh dấu để qua đó có thể viết CSS định dạng cho các phần tử trong văn bản HTML

47

Các thẻ (tag) đặc biệt : thẻ meta

❖ Mô tả thông tin về trang web một cách ngắn gọn

```
<html lang="vi">
    <head>
        <meta charset="UTF-8" />
        <meta name="viewport" content="width=device-width, initial-scale=1.0" />
        <meta name="description" content="Free Web tutorials" />
        <meta name="keywords" content="HTML,CSS,XML,JavaScript" />
        <meta name="author" content="Hoàng Sơn" />
    </head>
<body>
    ...
</body>
</html>
```

48

Các thẻ (tag) đặc biệt : thẻ script

❖ **Khai báo các phương thức xử lý phía Client**

✓ Client Script: JavaScript, VBScript

❖ **Cú pháp**

```
<script type="text/javascript">
    <!-- Các hàm hoặc phương thức (mặc định hoặc do người lập trình định nghĩa) -->
    document.write("Hello World!")
</script>
```

➢ **Các thuộc tính**

- **type** : xác định loại script nào được sử dụng
  "text/javascript" hoặc "text/vbscript"
- **src** : đường dẫn của file script được sử dụng
  `<script type="text/javascript" src="myScript.js"></script>`

49

Các thẻ (tag) đặc biệt : thẻ link

❖ **Thường dùng để kết nối với file định dạng css**

❖ **Cú pháp**

```
<head>
…
    <link rel="stylesheet" type="text/css" href="theme.css" />
…
</head>
```

50

<!DOCTYPE>

❖ **Xác định phiên bản HTML đang sử dụng**

❖ **Cho biết các quy định (luật) của ngôn ngữ đánh dấu**

Ví dụ:

```
<!DOCTYPE html>
```

✓ DOCTYPE trên xác định phiên bản HTML5

51

<!DOCTYPE>

Ví dụ:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01
Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
```

✓ DOCTYPE trên xác định phiên bản HTML 4.01, trong văn bản HTML được phép sử dụng các thẻ đánh dấu dạng trình bày (<u>, <font>…), không sử dụng frame

52

CSS

53

CSS – Khái niệm

- Cascading Style Sheets
- Xác định cách thức trình bày các phần tử HTML
- Cú pháp : selector {proterty:value;}
  - selector : thẻ HTML, . , #

Ví dụ: body, p , h1 , …

.myclass

#myid

input[type="text"]

#myid img

div .myclass

*

a:link

a:hover

a:visited

a:active

54

CSS – Khái niệm

- property : thuộc tính quy định cách trình bày

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

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

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

☞ *Chú thích trong CSS : dùng cặp /* … */*

55

CSS3 – Tiền tố

Tiền tố CSS vendor là một cách viết CSS cho một loại trình duyệt cụ thể ở các phiên bản thấp. **Hiện tại** các trình duyệt đều hỗ trợ thống nhất các thuộc tính CSS3 nên **không cần thiết thêm tiền tố**.

-moz-	(Firefox)
-o-	(Opera)
-ms-	(Internet Explorer)
-webkit-	(Safari, Chrome)

56

CSS3 – Tiền tố

```
#mybox{
      border: solid 1px red;
      border-radius: 5px 10px 5px 10px;
      -moz-border-radius: 5px 10px 5px 10px;
      -webkit-border-radius: 5px 10px 5px 10px;
      -ms-border-radius: 5px 10px 5px 10px;
      -o-border-radius: 5px 10px 5px 10px;
}
```

57

Đơn vị CSS

❖ Đơn vị kích thước tương đối

Đơn vị	Mô tả
%	% so với thành phần chứa đối tượng
vw	% của chiều rộng cửa sổ khung hình
vh	% của chiều cao cửa sổ khung hình
vmin	% của chiều khung nhìn nhỏ nhất
vmax	% của chiều khung nhìn lớn nhất
em	kích cỡ của font hiện tại (font-size) đối tượng hoặc thành phần chứa đối tượng
rem	giá trị tương đối với font của thành phần gốc (html)
ex	chiều cao của 1 chữ x (in thường) của font hiện tại
ch	chiều rộng của số 0

58

Đơn vị CSS

❖ Đơn vị kích thước tuyệt đối

Đơn vị	Mô tả
px	Pixel
pt	point (1pt = 1/72 in)
cm	centimeter
mm	millimeter
in	inch
pc	pica (1pc = 12pt)

59

Đơn vị CSS

❖ Đơn vị màu sắc

Đơn vị	Mô Tả
Color-name	Tên màu tiếng Anh. *Ví dụ*: black, white, red, green, blue, cyan, magenta,…
Hexadecimal RGB	Mã màu RGB dạng hệ thập lục. *Ví dụ*: #FFFFFF: trắng, #000000: đen, #FF00FF: đỏ tươi.
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.

60

CSS

➢ **Đơn vị góc (CSS3)**

Đơn vị	Mô Tả
deg	góc độ (1 vòng là 360deg)
rad	góc radian (1 vòng là 2π)
grad	độ dốc (1 vòng là 400grad)
turn	độ xoay (1 vòng là 1turn)

61

CSS

➢ **Đơn vị thời gian (CSS3)**

Đơn vị	Mô Tả
ms	milliseconds
s	seconds

62

CSS

➢ **Đơn vị tần số (CSS3)**

Đơn vị	Mô Tả
Hz	Hertz
kHz	kilohertz

63

Khai báo code CSS

➢ **Kiểu thuộc tính (inline style):** không cần selector

Ví dụ: `<body style="background-color:#FFF;">`

➢ **Khai báo trong thẻ <style> (Internal Style Sheet)**

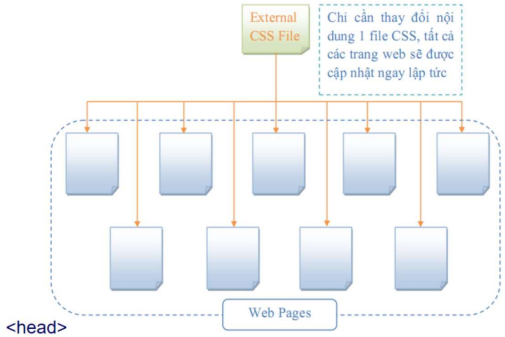
Ví dụ:

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

64

Khai báo code CSS

➢ **Kiểu bên ngoài (liên kết với file CSS bên ngoài)**




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

65

CSS

☞ **Trong CSS chúng ta còn có thể sử dụng @import để nhập một file CSS vào CSS hiện hành.**

Cú pháp: *@import url(link)*

Ví dụ:

```
<style type="text/css">
    @import url("import3.css");
    @import url("import2.css") screen;
    @import url("import2.css") print;

    p { color : #F00; }
</style>
```

66

CSS

➢ **Độ ưu tiên**

- ✓ Inline Style
- ✓ Internal Style Sheet
- ✓ External Style Sheet
- ✓ Browser default

Tăng dần

☞ **Để thay đổi độ ưu tiên thì sử dụng thuộc tính !important**

Ví dụ:

```
p {
    background-color:#FF0000;
    text-align:left !important;
    border:1px solid #FF0000;
    color:#333 !important
}
```

67

CSS - Background

❖ **Màu nền**

```
p {
    background-color:red;
}

h1 {
    background-color:#FF9966;
}
```

❖ **Ảnh nền**

```
body {
    background-image:url(../pictures/bgr.jpg)
}
```

68

CSS - Background

➢ **Lặp lại ảnh nền :** sử dụng trong trường hợp ảnh nền quá nhỏ

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

```
body {
    background-image:url(../pictures/bgr.jpg);
    background-repeat:no-repeat;
}
```

69

CSS - Background

➢ **Khóa ảnh nền :** thuộc tính **background-attachment**

- **scroll :** Ảnh nền sẽ cuộn cùng nội dung trang web, là giá trị mặc định.
- **fixed** : Cố định ảnh nền so với nội dung trang web. Khi áp dụng giá trị này, ảnh nền sẽ đứng yên khi cuộn trang web.

➢ **Định vị ảnh nền :** thuộc tính **background-position**

Ví dụ:

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

70

CSS - Background

❖ **Rút gọn**

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

Ví dụ:

```
body {
    background:transparent url(../pictures/bgr.jpg) no-repeat fixed right bottom;
}
```

71

CSS3 – Background

❖ **Có thể áp dụng đa hình nền**

```
#example {
  background: url(img_1.jpg) left top no-repeat,
              url(img_2.jpg) right bottom no-repeat,
              url(img_3.jpg) no-repeat center fixed;
  background-size: 50px 80px, auto, cover;
                   /* cover: tràn màn hình */
}
```

72

CSS3 – Background Gradient

❖ **Linear Gradient**

```
#gradDefault {    /* Mặc định màu phối từ trên xuống */
   background: linear-gradient(red, yellow);
}

#grad1 {
   background: linear-gradient(to top right, red, yellow);
}

#grad2 {    /* Đa màu */
   background: linear-gradient(to bottom left, red, yellow,
                               blue, #CFF0AD);
}
```

73

CSS3 – Background Gradient

❖ **Linear Gradient**

```
#grad3 {    /* Dùng độ quay */
   background: linear-gradient(-90deg, red, yellow);
}

#grad4 {    /* Sử dụng độ mờ của màu */
   background: linear-gradient(to right, rgba(255,0,0,0.2),
                                rgba(0,255,0,0.8));
}

#grad5 {    /* Lặp màu theo tỷ lệ*/
   background: repeating-linear-gradient(45deg, red, yellow 10%);
}
```

74

CSS3 – Background Gradient

❖ **Radial Gradient**

```
#grad6 {    /* Mặc định ellipse */
   background: radial-gradient(red, yellow);
}

#grad7 {    /* Phối tròn có tỷ lệ */
   background: radial-gradient(circle, red 30%, yellow);
}

#grad8 {    /* Lặp màu theo tỷ lệ */
   background: repeating-radial-gradient(yellow, green 10%,
               rgba(255,0,0,0.3) 20%);
}
```

75

CSS - Font

❖ **font-family** : định nghĩa một danh sách ưu tiên các font sẽ được dùng để hiển thị một thành phần trang web.

Ví dụ:

```
body { font-family:"Times New Roman",Tahoma,sans-serif ; }

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

❖ **font-style**

Ví dụ:

```
p { font-style:italic; }          /* in nghiêng */
a { font-style:oblique; }         /* kiểu in nghiêng khác */
h1 { font-style:normal; }         /* kiểu in thường */
```

76

CSS - Font

❖ **font-variant :** chọn giữa chế độ bình thường và small-caps của một font chữ (font small-caps là font sử dụng chữ in hoa có kích cỡ nhỏ hơn in hoa chuẩn để thay thế những chữ in thường)

Ví dụ:

```
p { font-variant:small-caps; }
```

❖ **font-weight**

Ví dụ:

```
p { font-weight:bold; }
h1 { font-weight:bolder; }
#myid { font-weight:900; }
```

77

CSS - Font

❖ **font-size**

Ví dụ:

```
p { font-size:150%; }
h1 { font-size:3em; }
#myid { font-size:30px; }
.myclass { font-size:xx-large; }
```

78

CSS - Font

❖ **Rút gọn**

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

Ví dụ:

```
p {
    font: italic bold 120% arial,verdana,sans-serif;
}
```

79

CSS3 – Web Fonts

❖ **Nhúng font chữ vào trang web (trong trường hợp font không phổ biến)**

```
@font-face {
    font-family: myFirstFont;
    src: url(sansation_bold.woff);
    font-weight: bold;
}

#mytext {
    font-family: myFirstFont;
}
```

80

CSS - Text

❖ **color :** màu chữ

Ví dụ:

```
body { color:#000; }
    p { color: red; }
```

❖ **text-indent :** tạo tab

Ví dụ:

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

❖ **Màu chữ gradient**

Ví dụ:

```
span {
    color:transparent;
    background:linear-gradient(red, yellow);
    background-clip:text;
    -webkit-background-clip:text;
}
```

81

CSS - Text

❖ **text-align:** left | right | center| justify

Ví dụ:

```
p { text-align:justify; }
```

❖ **letter-spacing:** khoảng cách giữa các ký tự

Ví dụ:

```
p { letter-spacing:5px; }
```

82

CSS - Text

❖ **text-decoration:** thêm hiệu ứng

Ví dụ:

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

❖ **text-transform:** định dạng chữ in hoa hay thường

Ví dụ:

```
h1 {text-transform:uppercase;}   /* IN HOA */
h2 {text-transform:lowercase;}   /* in thường*/
h3 {text-transform:capitalize;}  /* In Hoa Đầu Mỗi Từ */
```

83

CSS3 – Hiệu ứng văn bản

❖ **text-overflow:** cắt ngắn đoạn văn bản để khớp với kích thước đã cài đặt.

```
div.test {
  width: 200px;
  white-space: nowrap;  /* phần văn bản thừa không xuống hàng */
  overflow: hidden;     /* ẩn phần văn bản thừa */
  text-overflow: ellipsis;  /* đánh dấu ... văn bản thừa */
}

div.test:hover {  /* hiển thị văn bản thừa khi trỏ vào */
    text-overflow: inherit;
    overflow: visible;
}
```

84

CSS3 – Hiệu ứng văn bản

❖ **word-wrap:** cho phép từ quá dài bị cắt để xuống dòng

```
p {
  width: 50px;
  word-wrap: break-word;
}
```

85

CSS3 – Đổ bóng

❖ **text-shadow:** (ngang) (dọc) [độ mờ] [màu bóng];

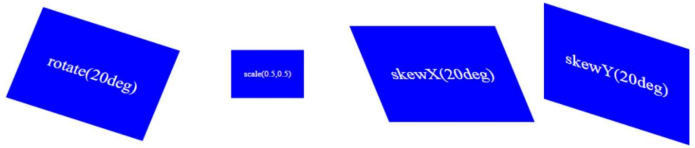
```
h1 {
  color: red;
  text-shadow: 2px 2px 4px rgba(255,80,5,0.5);
}
```

❖ **box-shadow:** (ngang) (dọc) [độ mờ] [màu bóng] [kiểu];

```
#box1 {
  background: red;
  box-shadow: 5px 5px 40px yellow inset;
  /* inset: đổ bóng phía trong, mặc định đổ bóng ngoài */
}
```

86

CSS3 – Transform




```
.box1 {     /* Xoay 20⁰ */
   transform: rotate(20deg);
}

.box2 { /* Phóng to – thu nhỏ */
   transform: scale(0.5, 0.5);
}
```

```
.box3 {     /* Nghiêng X */
   transform: skewX(20deg);
}

.box4 {     /* Nghiêng Y */
   transform: skewY(20deg);
}
```

87

CSS - Border

❖ **border-width:** quy định độ rộng cho viền của một đối tượng

Ví dụ:

```
p {
    border-width:thin medium thick 10px;
}
```

- ✓ Viền trên mảnh
- ✓ Viền phải vừa
- ✓ Viền dưới dày
- ✓ Viền trái 10px

❖ **border-color:** quy định màu viền của một đối tượng

88

CSS - Border

❖ **border-style:** solid, dotted, dashed, double, groove, ridge, inset và outset. Ngoài ra còn có none hay hidden dùng để ẩn viền

Viền Kiểu Solid | Viền Kiểu Dotted | Viền Kiểu Dashed | Viền Kiểu Double

Viền Kiểu Groove | Viền Kiểu Ridge | Viền Kiểu Inset | Viền Kiểu Outset

☞ *Viết rút gọn*

border: <border-style> |<border-width> |<border-color>

89

CSS3 – Border

❖ **border-radius** : tạo viền tròn cho phần tử HTML

```
#rcorners4 {
    border:solid 2px #73AD21;
    border-radius: 5px 10em 2% 20pt;
}

#rcorners2 {
    background: #73AD21;
    border-radius: 25px 30px;
}

#rcorners1 {
    background: url(bg.jpg);
    border-radius: 50%;
}
```

90

CSS3 – Border

❖ **border-image** : tạo viền cho phần tử HTML bằng hình vẽ

```
#borderimg1 {
    border:solid 10px transparent;
    border-image:url(border.png) 20 round;
}

#borderimg2 {
    border:solid 10px transparent;
    border-image: url(border.png) 30% stretch;
}
```

91

CSS3 – Border

❖ **border gradient**

```
#borderGrad {
    border:solid 10px transparent;
    border-image:linear-gradient(red, yellow);
    border-image-slice:1;
}
```

92

CSS - Width & Height

❖ **width:** quy định chiều rộng của một đối tượng

Ví dụ:

```
p {
    width:700px;
}
```

❖ **max-width:** quy định chiều rộng tối đa của một đối tượng

❖ **min-width:** quy định chiều rộng tối thiểu của một đối tượng

❖ **height, max-height, min-height**

93

CSS - Margin & Padding

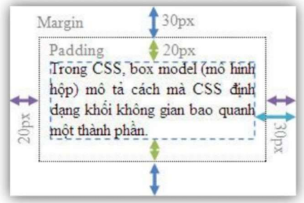
❖ **Định dạng khối không gian bao quanh một thành phần.**

Ví dụ: cho mã HTML

```
<p>
    Trong CSS, box model (mô hình hộp) mô tả cách mà CSS
    định dạng khối không gian bao quanh một thành phần.
</p>
```

Định dạng CSS

```
p {
    width:200px;
    margin:30px 20px;
    padding:20px 10px;
    border:1px dotted #000;
    text-align:justify
}
```



94

CSS - Margin & Padding

❖ **Margin**

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

hoặc **margin:**<margin-top&bottom> | <margin-right&left>

Ví dụ:

```
body {
    margin:80px 30px 40px 50px;
    border:1px solid #FF0000;
}
```

❖ **Padding**

(tương tự margin)

95

CSS3 – User Interface

❖ **box-sizing** : Bình thường kích thước của phần tử HTML sẽ là: (width, height) + padding + border. Để cài đặt chính xác kích thước theo (width, height) bỏ qua padding và border thì dùng box-sizing

```
.div2 {
    width: 300px;
    height: 100px;
    padding: 50px;
    border: 5px solid red;
    box-sizing: border-box;
}
```

96

CSS3 – User Interface

❖ **resize** : Quy định một phần tử HTML có thể hay không thể thay đổi kích thước bởi người dùng

```
.div1 {
    resize: horizontal;
    overflow: auto;
}

.div2 {
    resize: vertical;
    overflow: auto;
}

.div3 {
    resize: both;
    overflow: auto;
}

.div4 {
    resize: none;
}
```

97

CSS3 – User Interface

❖ **outline-offset** : tạo viền bên ngoài phạm vi phần tử HTML nhưng không chiếm không gian.

```
#ex1 {
    margin: 20px;
    border: 1px solid black;
    outline: 4px solid red;
    outline-offset: 15px;   /*Khoảng cách không gian*/
}
```

98

CSS3 – Chia cột văn bản

```
.newspaper {
  column-count: 3;    /* số cột */
  column-gap: 40px;   /* khoảng cách giữa các cột */
  column-rule: solid 1px blue;   /* đường kẻ phân cột */
}
```

✓ Nếu có đoạn văn bản trong vùng chia cột cần ra ngoài thì sử dụng column-span

```
#outColumn {
  column-span: all;
}
```

99

CSS3 – Hình vẽ

❖ **Kích thước tự động**

```
img {
  width: 50%;
  height: auto; /* Tự động chỉnh theo tỷ lệ với chiều rộng */
  opacity: 0.5; /* Độ mờ */
}
```

100

CSS3 – Hình vẽ

❖ **object-fit:** Thay đổi cách hiển thị của hình ảnh (video) trong vùng chứa.

```
#myImgDiv {
    width: 200px;
    height: 400px;
}

#myImgDiv img {    /* Hình ảnh trong vùng #myImgDiv */
    width: 400px;
    height: 500px;
}
```

101

CSS3 – Hình vẽ

```
#myImgDiv img  {object-fit: fill;}
/* Mặc định, tự động giãn hoặc co hình cho tràn
đầy vùng chứa */


#myImgDiv img  {object-fit: contain;}
/* tự động giãn hoặc co hình nhưng vẫn giữ tỷ lệ
của hình (sẽ có 1 chiều không tràn vùng chứa) */
```

102

CSS3 – Hình vẽ

```
#myImgDiv img  {object-fit: cover;}
/* Thường được sử dụng, tự động giãn hoặc co hình
nhưng vẫn giữ tỷ lệ của hình đồng thời tràn vùng
chứa cả 2 chiều → có thể mất 1 phần hình ảnh.

#myImgDiv img  {object-fit: scale-down;}
/* Tương tự như contain */

#myImgDiv img  {object-fit: none;}
/* Giữ nguyên kích thước hình ảnh ban đầu */
```

103

CSS - Float & Clear

❖ **float:** cố định một đối tượng về bên trái hay bên phải không gian bao quanh nó.

- left : Cố định phần tử về bên trái.
- right : Cố định phần tử về bên phải.
- none : Bình thường.

Ví dụ: chia văn bản thành 2 cột

```
.column1, .column2 {
    width:45%;
    float:left;
    text-align:justify;
    padding:0 20px;
}
```

```
.column1 {
    border-right:1px solid #000
}
```

104

CSS - Float & Clear

❖ **clear:** thường được gán vào các đối tượng liên quan tới đối tượng đã được float để quyết định cách chèn vào của các đối tượng

- left : tràn bên trái.
- right : tràn bên phải.
- both : tràn 2 phía
- none : bình thường.

105

CSS - Pseudo classes

❖ **Các hiệu ứng định dạng cho đối tượng liên kết ở một trạng thái xác định**

Ví dụ:

```
a:link {
    color:#00FF00;
}
a:hover {
    color:#FF00FF;
}
a:visited {
    color:#FF0000;
}
a:active {
    color:# 662D91;
}
```

106

CSS - Position

❖ **absolute position**

Ví dụ: đặt 4 ảnh ở 4 góc tài liệu

```
#myimg1 {
    position:absolute;
    top:20px;
    left:50px;
}

#myimg2 {
    position:absolute;
    top:20px;
    right:30px;
}

#myimg3 {
    position:absolute;
    bottom:0;
    left:50px;
}

#myimg4 {
    position:absolute;
    bottom:0;
    right:30px;
}
```

❖ **relative position :** định vị so với vị trí ban đầu

107

CSS - Layers

❖ **Cách đặt một thành phần này lên trên một thành phần khác.**

Ví dụ: đặt 4 ảnh chồng lên nhau

```
#myimg1 {
    position:absolute;
    top:20px;
    left:50px;
    z-index:1;
}

#myimg2 {
    position:absolute;
    top:20px;
    left:60px;
    z-index:2;
}

#myimg3 {
    position:absolute;
    top:20px;
    left:70px;
    z-index:3;
}

#myimg4 {
    position:absolute;
    top:20px;
    left:80px;
    z-index:4;
}
```

108

CSS3 – Transition

- Thay đổi định dạng CSS một cách linh động theo thời gian

```
.box1 {
    width: 100px;
    height: 100px;
    background: red;
    transition: width 2s, height 4s, border-radius 3s;
}

.box1:hover {
    width: 300px;
    height: 300px;
    border-radius: 20px;
}
```

109

CSS3 – User Interface

- @media : Xác định thiết bị áp dụng mã CSS

```
/* Áp dụng cho màn hình có chiều rộng LỚN hơn 480px */
@media screen and (min-width: 480px) {
    .thisBox {
        background-color: lightgreen;
    }
}

/* Áp dụng cho màn hình có chiều rộng NHỎ hơn 960px */
@media screen and (max-width: 960px) {
    .thisBox {
        background-color: pink;
    }
}
```

110

JAVASCRIPT

Company Logo

111

Nội dung

- Giới thiệu về Javascript
- Nhúng Javascript vào trang web
- Kiểu dữ liệu & Các cú pháp Javascript
- Xử lý sự kiện
- DOM HTML với Javascript
- Ví dụ

112

Giới thiệu JavaScript

- Là một ngôn ngữ kịch bản (script) để viết kịch bản cho phía client.
- Chia sẻ xử lý trong ứng dụng web. Giảm các xử lý không cần thiết trên server.
- Giúp tạo các hiệu ứng, tương tác cho trang web.

113

Khai báo

- Định nghĩa script trực tiếp trong trang html

```
<script type="text/javascript">
    <!--
  // Lệnh Javascript
     -->
</script>
```

- Nhúng sử dụng script cài đặt từ 1 file .js khác

```
<script type="text/javascript" src="xxx.js">
</script>
```

114

Nhúng JavaScript vào trang web

```
<html>
  <head>
     <script type="text/javascript">
        some javascript statements
     </script>
  </head>
  <body>
     <script type="text/javascript">
        some statements
     </script>
     <script src="Tên_file_script.js">method()</script>
     <script type="text/javascript">
       // gọi thực hiện các phương thức được định nghĩa
       // trong "Tên_file_script.js"
     </script>
  </body>
</html>
```

115

Nhúng JavaScript vào trang web

- **Đặt giữa tag <head> và </head>:** script sẽ thực thi ngay khi trang web được mở.
- **Đặt giữa tag <body> và </body>:** script trong phần body được thực thi khi trang web đang mở (sau khi thực thi các đoạn script có trong phần <head>).
- **Số lượng đoạn client-script chèn vào trang không hạn chế.**
- **Chú thích trong JavaScript**

```
// Chú thích trên 1 dòng
/* Chú thích trên
      nhiều dòng */
```

116

Nhúng JavaScript vào trang web

Ví dụ:

```
<html>
  <body>
      document.write("Hello world!");
      <script type="text/javascript">
          document.write("Hello world!");
      </script>
  </body>
</html>
```

Biến số

❖ Quy tắc đặt tên biến

- Tên biến gồm các chữ cái và số nhưng phải bắt đầu bằng ký tự chữ cái hoặc ký tự gạch dưới (_)
- Không bắt đầu bằng ký tự số.
- Không dùng các ký tự đặc biệt như: (, [, { , # , & …
- Không chứa khoảng trắng, tên biến phải gợi nhớ
- Không trùng với từ khoá của JavaScript
- Tên biến ***phân biệt chữ hoa và chữ thường***

Biến số

- ***Các từ khóa trong javascript***

abstract	eval	int	static
boolean	extends	interface	super
break	false	long	switch
byte	final	native	synchrinized
case	finally	new	this
catch	float	null	throw
char	for	package	throws
class	function	parseFloat	transient
const	goto	parseInt	true
continue	if	private	try
default	implements	protected	var
do	import	public	void
double	in	return	while
else	instanceof	short	with

Biến số

❖Khai báo biến

- Sử dụng từ khóa **var**

  var ***tên_biến*** **=** ***giá_trị*****;**

- Có thể không cần khai báo biến trước khi sử dụng.

Các kiểu dữ liệu

Kiểu dữ liệu	Ví dụ	Mô tả
Object	var listBooks = **new** Array(10) ;	Trước khi sử dụng, phải cấp phát bằng từ khóa **new**
String	"The cow jumped over the moon." "40"	Chứa được chuỗi unicode Chuỗi rỗng ""
Number	0.066218 12	Theo chuẩn IEEE 754
boolean	true / false	
undefined	var myVariable ;	myVariable = undefined
null	connection.Close();	connection = null

121

Các kiểu dữ liệu

- Biến tự đổi kiểu dữ liệu khi giá trị mà nó lưu trữ thay đổi

Ví dụ:

```
var x = 10;                 // x kiểu Number
x = "hello world !";        // x kiểu String
```

- Có thể cộng 2 biến khác kiểu dữ liệu

Ví dụ:

```
var x;
x = "12" + 34.5;     // KQ: x = "1234.5"
```

122

Toán tử trong JavaScript

- Toán tử số học

Toán tử	Chức năng	Ví dụ		Kết quả
+	Cộng	x=2;	x+2;	**4**
-	Trừ	x=2;	5-x;	**3**
*	Nhân	x=3;	x*3;	**9**
/	Chia	x=5;	x/2;	**2.5**
%	Chia lấy phần dư	x=7;	x%3;	**1**
++	Tăng 1	x=1;	x++;	**2**
--	Giảm 1	x=1;	x--;	**0**

123

Toán tử trong JavaScript

- Toán tử gán

Toán tử	Ví dụ	Tương đương
=	x=y;	x=y;
+=	x+=y;	x=x+y;
-=	x-=y;	x=x-y;
*=	x*=y;	x=x*y;
/=	x/=y;	x=x/y;
%=	x%=y;	x=x%y;

124

Toán tử trong JavaScript

❖ Toán tử so sánh

Toán tử	Chức năng
==	Bằng
!=	Không bằng (khác)
>	Lớn hơn
<	Nhỏ hơn
>=	Lớn hơn hoặc bằng
<=	Nhỏ hơn hoặc bằng

125

Toán tử trong JavaScript

❖ Toán tử logic

Toán tử	Chức năng	Ví dụ
&&	và	(x>1) && (y<2)
||	hoặc	(x==1) || (y==-1)
!	not	!(x==y)

126

Cấu trúc điều khiển trong JavaScript

❖ Rẽ nhánh if

➢ Rẽ nhánh đơn

```
if(biểu_thức_logic) {
    khối lệnh được thực hiện nếu biểu_thức_logic đúng;
}
```

➢ Rẽ nhánh if...else...

```
if(biểu_thức_logic) {
    khối lệnh được thực hiện nếu biểu_thức_logic đúng;
}
else {
    khối lệnh được thực hiện nếu biểu_thức_logic sai;
}
```

127

Cấu trúc điều khiển trong JavaScript

➢ Rẽ nhánh if...else... lồng nhau

```
if(biểu_thức_logic_1) {
    khối lệnh được thực hiện nếu biểu_thức_logic_1 đúng;
}
else if(biểu_thức_logic_2) {
    khối lệnh được thực hiện nếu biểu_thức_logic_2 đúng;
}
else {
    khối lệnh được thực hiện nếu biểu_thức_logic_2 sai;
    // đồng nghĩa với biểu_thức_logic_1 cùng sai
}
```

128

Cấu trúc điều khiển trong JavaScript

➢ **Sử dụng toán tử điều kiện**

var **x** = (***biểu_thức_logic***) **? value_true : value_false;**

/ x sẽ được gán giá trị value_true nếu biểu_thức_logic đúng, ngược lại nếu biểu_thức_logic sai thì x sẽ được gán giá trị value_false */*

Ví dụ:

```
var x = prompt("Nhập vào giá trị của x:");
if (!isNaN(x)) {
    y = parseFloat(x);
    s = (y == 0) ? "số 0" : "số khác không";
    document.write(x + " là " + s);
}
else {
    document.write(x + " không phải là số");
}
```


Cấu trúc điều khiển trong JavaScript

❖ **Rẽ nhánh switch**

```
switch (biểu_thức) {
   case giá_trị_biểu_thức_1 :
      khối lệnh 1; break;
   case giá_trị_biểu_thức_2 :
      khối lệnh 2; break;
   ...
   default :
      khối lệnh khi biểu_thức trả về giá trị còn lại;
}
```


Cấu trúc điều khiển trong JavaScript

Ví dụ:

```
var t = prompt("Nhập tháng:");
switch (parseInt(t)) {
    case 1: case 3: case 5: case 7: case 8 : case 10: case 12:
        alert("Tháng "+ t+ " có 31 ngày"); break;
    case 2:
        alert("Tháng "+t + " có 28 ngày"); break;
    case 4: case 6: case 9: case 11:
        alert("Tháng "+t +" có 30 ngày"); break;
    default:
        alert("Nhập sai");
}
```


Cấu trúc điều khiển trong JavaScript

❖ **Vòng lặp for**

```
for(giá_trị_khởi_tạo; biểu_thức_logic; biểu_thức_thay_đổi_giá_trị) {
   Khối lệnh;
}
```

```
for(chỉ_số  in tên_mảng) {
    Khối lệnh;
}
```

Cấu trúc điều khiển trong JavaScript

Ví dụ:

```
var   a = new Array("An","Bình","Châu")
for(var i=0;i < a.length;i++){
   document.write(a[i] + "<br />");
}
```

Ví dụ:

```
var   a = new Array("An","Bình","Châu")
for(var i in a){
   document.write(a[i] + "<br />");
}
```

133

Cấu trúc điều khiển trong JavaScript

Ví dụ:

```
function Sinhvien(){
   this.ten = "A";
   this.tuoi = 20;
}

var sv = new Sinhvien;
for(var attr in sv){
   document.write(attr + "<br />" );
}
```

134

Cấu trúc điều khiển trong JavaScript

❖ **Vòng lặp while**

```
while (biểu_thức_logic) {
   khối lệnh thực hiện;
}

do {
   khối lệnh thực hiện;
} while (biểu_thức_logic)
```

(Khối lệnh trong vòng lặp sẽ thực hiện cho đến khi nào biểu_thức_logic có giá trị false)

135

Cấu trúc điều khiển trong JavaScript

➢ *Có thể dùng lệnh break hay continue để dừng hoặc tiếp tục vòng lặp*

Ví dụ:

```
var   a = new Array("An","Bình","Châu")
for(var i in a){
   document.write(a[i] + "<br />");
   if (a[i] == "Bình"){ break; }
}
```

136

Cấu trúc điều khiển trong JavaScript

❖ **Quản lý lỗi**

```
try {
   Khối lệnh cần thực hiện có thể gây lỗi;
}
catch (error) {
   Khối lệnh cần thực hiện trong trường hợp có lỗi;
}
finally {
   Khối lệnh luôn được thực hiện;
}
```

137

Cấu trúc điều khiển trong JavaScript

Ví dụ:

```
try{
   myFunction();
   alert("Hàm có tồn tại");
}
catch(e){
   alert("Có lỗi : " + e.message);
}
finally{
   alert("Chú ý cẩn thận!");
}
```

138

Hàm trong JavaScript

❖ **Khai báo**

```
function Tên_hàm(thamso1, thamso2,..) {
    .....
    return <value>;
}
```

139

Hàm trong JavaScript

Ví dụ:

```
<html>
   <head><title>Function</title></head>
<body>
   <script>
      function sum(a , b) {
         c = a + b;
         document.write(c);
         return;
      }

      sum(2,3);
   </script>
</body>
</html>
```

140

Hàm trong JavaScript

☞ ***Chú ý*** : **Có thể khai báo hàm không tên** *(thường dùng cho các sự kiện)*

Ví dụ:

```
<script type="text/javascript">
    document.onkeypress = function(e) {
        alert("Một phím trên bàn phím vừa được nhấn");
    }
</script>
```

141

Hàm trong JavaScript

❖ **Một số quy tắc chung**

- Khối lệnh được bao trong dấu { }
- Mỗi lệnh nên kết thúc bằng dấu ;
- Cách ghi chú thích:

```
// Chú thích 1 dòng
/* Chú thích
   nhiều dòng */
```

142

Một số hàm trong JavaScript

❖ **alert(*"nội dung thông báo"*) :**

Dùng hiển thị một hộp thông báo có nút OK

Ví dụ:

```
<script>
    alert("Hello World");
</script>
```

143

Hàm trong JavaScript

❖ **prompt(*"nội dung đối thoại"*,*giá trị khởi tạo*)**

Tạo hộp thoại chứa 2 nút OK ,Cancel và một textbox để người dùng nhập nội dung, giá trị trả về của hàm prompt là nội dung nhập trong textbox

Ví dụ:


```
<script>
   a = prompt("Your Lastname:","");
   b = prompt("Your FirstName:","");
   document.write("Your FullName is :"+ a + " " + b)
</script>
```

144

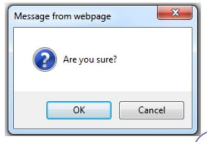
Hàm trong JavaScript

❖ **confirm(*"Nội dung thông báo"*)**

Hiển thị hộp thông báo có 2 nút OK và Cancel. Hàm trả về giá trị true nếu người sử dụng click OK và ngược lại thì trả về giá trị false.

Ví dụ:

```
<script>
    a = confirm("Are you sure?");
    //...
</script>
```



145

Hàm trong JavaScript

❖ **eval(*"Biểu thức"*)**

Chuyển đổi một chuỗi thành số hoặc giá trị logic (*chuỗi có thể chứa biểu thức toán học*).

Ví dụ:

```
x = 1;
y = eval("x*2 + 3");        // y = 5
y = eval("x > 2");          // y = false
y = eval("123.456");        // y = 123.456
y = eval("ABC");
```

146

Hàm trong JavaScript

❖ **parseInt(*strNum*)**

- Trả về một số nguyên từ chuỗi *strNum*.
- Nếu strNum theo sau là ký tự chữ thì các ký tự này sẽ bị bỏ qua.
- Nếu strNum không bắt đầu bằng số thì hàm này trả về giá trị ***NaN*** (Not a Number)

Ví dụ:

```
x = parseInt("123.456")      // x = 123
x = parseInt(".456")         // x = NaN
```

147

Hàm trong JavaScript

❖ **parseFloat(*strNum*)** - Tương tự như ***parseInt*** nhưng trả về số thực

Ví dụ:

```
x = parseFloat ("123.456")      // x = 123.456
x = parseFloat(".456")          // x = 0.456
x = parseFloat("123.4a56")      // x = 123.4
x = parseFloat("a123.456")      // x = NaN
```

148

Hàm trong JavaScript

❖ **isNaN(*str*)**

Trả về *false* nếu str là chuỗi số, ngược lại là true

Ví dụ:

```
c = isNaN("123.456")        // c = false
c = isNaN(".123")           // c = false
c = isNaN("123-456")        // c = true
```

149

Hàm trong JavaScript

❖ **setTimeout(*"Lệnh hoặc hàm JavaScript"*, *thời gian*)**

- Thực hiện một lệnh (hàm) JavaScript sau một khoảng thời gian nào đó.
- Hàm trả về một ID (duy nhất đối với mỗi hàm setTimeout thực hiện một lệnh). Giá trị ID này dùng để xoá thời gian đã thiết lập bằng hàm clearTimeout

Ví dụ:

```
idq = setTimeout("alert('Goodbye!')",1000) ;
        // Sau 1000ms  thì hiện thông báo
```

❖ **clearTimeout(*IDTime*) :** hủy thời gian đã thiết lập bởi setTimeout

150

Hàm trong JavaScript

❖ **setInterval(*"Lệnh hoặc hàm JavaScript"*, *thời gian*)**

❖ **clearInterval(*IDTime*) :** hủy thời gian đã thiết lập bởi setInterval

(Sử dụng tương tự như setTimeout và clearTimeout)

151

Đối tượng trong JavaScript - String

❖ **String**

- **Khai báo**

```
var     st1 = "Đây là string 1";
/* hoặc */    st2 = "Đây là string 2";
/* hoặc */    st3 = new String("Đây là string 3");
```

- **Ghép chuỗi**

```
var     st4 = st1 + st2 + st3 + " là chuỗi ghép";
```

152

Đối tượng trong JavaScript - String

➢ **Một số ký tự đặc biệt**

- **\n** : xuống dòng
- **\t** : tab
- **\b** : BackSpace
- **\&** : dấu &
- **\"** : dấu "
- **** : dấu \

153

Đối tượng trong JavaScript - String

➢ **Các phương thức của đối tượng String**

- **length** : cho biết số lượng ký tự có trong chuỗi

  Ví dụ:

```
var   st = "ABC abc";
var   x = st.length;        // x = 7
```

- **charAt(pos)** : cho biết ký tự ở vị trí *pos* trong chuỗi

  Ví dụ:

```
var   st = "ABC abc";
var   p = st.charAt(2);     // p = "C"
```

154

Đối tượng trong JavaScript - String

- **concat(s1,s2,...)** : ghép các giá trị *s1, s2,...* vào cuối chuỗi gốc

  Ví dụ:

```
var   st = "ABC abc";
var   s = st.concat("X",1,2);
      // s = "ABC abcX12"
```

- **substr(start,*numberOfchar*)** : lấy ra chuỗi con bắt đầu từ vị trí *start (vị trí đầu tiên là 0)* với *numberOfchar* ký tự

  Ví dụ:

```
var   st = "ABC abc";
var   s = st.substr(1,2);     // s = "BC"
```

155

Đối tượng trong JavaScript - String

- **indexOf(subStr,*start*)** : tìm chuỗi con *subStr* bắt đầu từ vị trí *start.*

  (kết quả trả về vị trí của ký tự đầu tiên của chuỗi con, nếu không tìm thấy thì trả về giá trị -1)

  Ví dụ:

```
var   st = "ABC abc";
var   x = st.indexOf("abc");      // x = 4
var   y = st.indexOf("abc",5);    // y = -1
```

156

Đối tượng trong JavaScript - String

- **toLowerCase() :** chuyển các ký tự trong chuỗi thành chữ thường

Ví dụ:

```
var   st = "ABC abc";
var   s = st.toLowerCase();
      // s = "abc abc"
```

- **toUpperCase() :** chuyển các ký tự trong chuỗi thành chữ hoa

Ví dụ:

```
var   st = "ABC abc";
var   s = st.toUpperCase();
      // s = "ABC ABC"
```

157

Đối tượng trong JavaScript - String

- **match(st) :** trả về thông tin giống nhau giữa biểu thức so mẫu *st* và chuỗi gốc

Ví dụ:

```
var   st = "ABC abc";
var   s = st.match("abc");        // s = "abc"
```

- **replace(old, new) :** thay thế chuỗi con *old* (trong chuỗi gốc) bằng chuỗi *new*

Ví dụ:

```
var   st = "ABC abc";
var   s = st.replace("abc","xyz");
      // s = "ABC xyz"
```

158

Đối tượng trong JavaScript - String

- **split(symbol,*maxItem*) :** tách chuỗi thành 1 mảng tối đa *maxItem* phần tử với mỗi phần tử là các chuỗi con phân biệt bởi ký tự *symbol* trong chuỗi gốc

Ví dụ:

```
var          st = "ABC-abc-XYZ-123";
var itemArray = new Array() ;
    itemArray = st.split("-");
// itemArray = ["ABC","abc","XYZ","123"]
```

- ***regx*.test(s) :** kiểm tra các ký tự chuỗi gốc s có phù hợp với các ký tự được liệt kê trong tập hợp *regx*

159

Đối tượng trong JavaScript - Array

❖ **Array**

➢ **Khai báo mảng 1 chiều**

```
      var   arr1 = new Array();
/* hoặc */ var   arr2 = new Array(10);
/* hoặc */       arr3 = new Array(1,2,"ABC","xyz",true);
/* hoặc */       arr4 = [1,2,"ABC","xyz",true];
```

✓ ***Sử dụng***

```
x = arr4[0] + arr4[1];        // x = 3;
```

160

Đối tượng trong JavaScript - Array

➢ **Khai báo mảng đa chiều**

arr5 = **[** [1,2] **,** ["ABC","xyz"] **,** true**]**;

✓ ***Sử dụng***

x = arr5**[0][0]** + arr5**[0][1]**; **// x = 3;**

161

Đối tượng trong JavaScript - Array

✓ **Thuộc tính length**

Ví dụ: thêm 1 phần tử "END" vào cuối mảng

arr4[arr4.length] = "END";

➢ **Các phương thức của đối tượng Array**

- **toString()** hoặc **valueOf() :** cho kết quả là chuỗi liệt kê các phần tử của mảng.

Ví dụ:

```
var a = [1,2,"ABC"];
var s = a.toString();   // s = "1,2,ABC"
```

162

Đối tượng trong JavaScript - Array

- **concat(item1, item2,...) :** thêm phần tử vào ***cuối*** mảng *(các phần tử có thể là mảng)*

Ví dụ:

```
var a = [1,2,"ABC"];
arr = a.concat(0,["X","Y"],true);
// arr = [1,2,"ABC",0,["X","Y"],true]
```

163

Đối tượng trong JavaScript - Array

- **pop():** loại bỏ và trả về phần tử ***cuối*** cùng của mảng

Ví dụ:

```
var a = [1,2,"ABC"];
s = a.pop();
// a = [1,2]
// s = "ABC"
```

- **shift():** loại bỏ và trả về phần tử ***đầu*** tiên của mảng

164

Đối tượng trong JavaScript - Array

- **slice(start,*end*):** tạo mảng mới bắt đầu từ phần tử *start* đến trước phần tử *end* *(không thay đổi mảng gốc)*

Ví dụ:
```
var   a = [1,2,"ABC",true];
      arr = a.slice(1,3)          // arr = [2,"ABC"]
```

- **splice(start,numberOfItems,*item1,item2,...*):** tách mảng bắt đầu từ phần tử *start* lấy *numberOfItems* phần tử, bổ sung thêm các phần tử *item1*, *item2*,... *thế vào vị trí* các phần tử mất đi ở mảng gốc.

Ví dụ:
```
var   a = [1,2,"ABC",true];
      arr = a.splice(1,2,"X")     // arr = [2,"ABC"]
      // a = [1,"X",true]
```

165

Đối tượng trong JavaScript - Array

- **sort():** sắp xếp mảng 1 chiều theo thứ tự ***tăng*** dần *(thứ tự sắp xếp kiểu phần tử : number → string → boolean)*

Ví dụ:
```
var   a = [3,1,"X","A",true];
      a.sort();          // a = [1,3,"A","X",true]
```

- **reverse():** đảo ngược thứ tự các phần tử trong mảng

Ví dụ:
```
var   a = [3,1,"X","A",true];
      a.reverse();   // a = [true,"X","A",3,1]
```

166

Đối tượng trong JavaScript - Array

- **join(*symbol*):** ghép các phần tử của mảng vào 1 chuỗi phân biệt bởi *symbol*, mặc định là dấu **,**

Ví dụ:
```
var   a = [3,1,"X","A",true];
      s1 = a.join();          // s1 = "3,1,X,A,true"
      s2 = a.join("-");       // s2 = "3-1-X-A-true"
```

167

Đối tượng trong JavaScript - Date

❖ **Date**

➢ **Khai báo**

```
var   d1 = new Date();
/* hoặc */   d2 = new Date(2012,4,16);
/* hoặc */   d3 = new Date("May 16 2012");
```

☞ ***Chú ý***

- ✓ **Ngày : 1-31**
- ✓ **Tháng : 0 = January, 1 = February,...**
- ✓ **Thứ : 0 = Sunday, 1 = Monday,...**

168

Đối tượng trong JavaScript - Date

➢ **Các phương thức của đối tượng Date**

Phương thức	Miêu tả
getDate()	Trả về ngày của đối tượng Date (từ 1-31)
getDay()	Trả về thứ trong tuần của đối tượng Date (từ 0-6. 0=Sunday, 1=Monday, ...)
getMonth()	Trả về giá trị tháng của đối tượng Date (từ 0-11. 0=January, 1=February, ...)
getFullYear()	Trả về giá trị năm của đối tượng Date (4 chữ số)
getYear()	Trả về giá trị năm của đối tượng Date (2 chữ số nếu năm < 2000).
getHours()	Trả về giá trị Giờ
getMinutes()	Trả về giá trị Phút
getSeconds()	Trả về giá trị Giây
toString()	Trả về kiểu chuỗi của đối tượng Date

169

Đối tượng trong JavaScript - Date

Phương thức	Miêu tả
setDate(value)	Thiết lập giá trị ngày (từ 1-31)
setMonth(value)	Thiết lập giá trị tháng (từ 0-11)
setYear(value)	Thiết lập giá trị năm
setFullYear(value)	Thiết lập giá trị năm
setHours(value)	Thiết lập giá trị giờ (0-23)
setMinutes(value)	Thiết lập giá trị phút (0-59)
setSeconds(value)	Thiết lập giá trị giây (0-59)

170

Đối tượng trong JavaScript - Date

Ví dụ:

```
var   d = new Date();
      d.setDate(16);
      d.setMonth(4);        // tháng 5
      d.setYear(2012);
myDate = d.getDate();       // myDate = 16
myMonth = d.getMonth();     // myMonth = 4
myYear = d.getYear();       // myYear = 2012
```

171

Đối tượng trong JavaScript - Math

✓ **Các thuộc tính**

Thuộc tính	Miêu tả	Giá trị
E	Số e	2.718281828459045
LN2	Logarit cơ số e của 2	0.6931471805599453
LN10	Logarit cơ số e của 10	2.302585092994046
LOG2E	Logarit cơ số 2 của e	1.4426950408889634
LOG10E	Logarit cơ số 10 của e	0.4342944819032518
PI	Số Pi	3.141592653589793
SQRT1_2	Căn bậc 2 của 1/2	0.7071067811865476
SQRT2	Căn bậc 2 của 2	1.4142135623730951

Ví dụ:

```
x = Math.PI;
```

172

Đối tượng trong JavaScript - Math

➢ Các phương thức của đối tượng Math

Phương thức	Miêu tả	Ví dụ
abs(x)	Lấy trị tuyệt đối của x	z = Math.abs(-5); // z = 5
random()	Lấy số ngẫu nhiên >=0 và <1	z = Math.random();
max(num1,num2,...)	Lấy số lớn nhất trong các số num1,num2,...	z = Math.max(9,3,8,4); // z = 9
min(num1,num2,...)	Lấy số nhỏ nhất trong các số num1,num2,...	z = Math.min(9,3,8,4); // z = 3
ceil(x)	Số nguyên nhỏ nhất >=x	z = Math.ceil(5.3); // z = 6
floor(x)	Số nguyên lớn nhất <=x	z = Math.floor(5.3); // z = 5
pow(x,y)	x^y (x mũ y)	z = Math.pow(2,4); // z = 16
round(x)	Làm tròn nguyên của số x	z = Math.round(10.3); // z = 10

173

Đối tượng trong JavaScript - Math

Phương thức	Miêu tả
exp(x)	e^x (e lũy thừa x)
log(x)	lnx (logarit cơ số e của x)
sqrt(x)	Căn bậc 2 của x
sin(x)	Sin
cos(x)	Cos
tan(x)	Tan
acos(x)	Arccos
asin(x)	Arcsin
atan(x)	Arctan
atan2(x,y)	Arctan của x/y

174

Đối tượng trong JavaScript - Number

➢ Các phương thức của đối tượng Number

- **toFixed(d)** : định dạng số với *d* chữ số sau dấu .

Ví dụ: x = 0.12345;

y = x.toFixed(2); *// y = 0.12*

- **toPrecision(n)** : định dạng số có *n* chữ số (kể cả phần lẻ)

Ví dụ: x = 0.12345;

y = x.toPrecision(4); *// y = 0.1234*

// nếu phần nguyên là 0 thì sẽ không được tính

- **toString()** : chuyển số thành chuỗi

175

Đối tượng trong JavaScript - window

➢ *Các thuộc tính của đối tượng window*

Thuộc tính	Miêu tả	Giá trị
status	Thiết lập thông báo tại thời điểm hiện hành	Text
location	Xác định vị trí trang hiện tại trong cửa sổ	Text URL
alwaysLowered	Hiển thị cửa sổ bên dưới các của sổ khác	true/false
alwaysRaised	Hiển thị cửa sổ trên tất cả các cửa sổ khác	true/false
fullscreen	Hiển thị chế độ đầy màn hình	true/false
height	Thiết lập chiều cao của cửa sổ	Số nguyên
width	Thiết lập chiều rộng của cửa sổ	Số nguyên
left	Thiết lập khoảng cách từ text đến cạnh cửa sổ	Số nguyên

176

Đối tượng trong JavaScript - window

Thuộc tính	Miêu tả	Giá trị
menubar	Hiển thị thanh menu bar	true/false
toolbar	Hiển thị thanh công cụ	true/false
titlebar	Hiển thị thanh tiêu đề	true/false
status	Hiển thị thanh trạng thái	true/false
resizable	Cho phép thay đổi kích thước cửa sổ	true/false
scrolbars	Cho phép xuất hiện thanh cuộn	true/false
closed	Trả về giá trị true, false. True khi cửa sổ đóng	true/false

177

Đối tượng trong JavaScript - window

➢ ***Các phương thức của đối tượng window***

Phương thức	Mô tả
close()	Đóng cửa sổ hiện hành
focus()	Đặt focus cho cửa sổ hiện hành
blur()	Bỏ focus cho cửa sổ hiện hành
open(*URL,name,specs, replace*)	**URL** : URL được nạp vào cửa sổ. **name**: tên cửa sổ . **specs** : danh sách các thuộc tính của cửa sổ: toolbars, menu, status ….(các thuộc tính phân cách bởi dấu , **replace**: đưa cửa sổ được mở vào history (*false*) hoặc thay thế history của cửa sổ hiện hành (*true*)

178

Đối tượng trong JavaScript - window

Phương thức	Mô tả
moveBy(dx,dy)	Di chuyển cửa sổ 1 đoạn vị trí (dx,dy) so với cửa sổ hiện hành
moveTo(x,y)	Di chuyển cửa sổ đến vị trí (x,y) so với màn hình
resizeBy(dx,dy)	Thay đổi kích thước cửa sổ (dx,dy) so với cửa sổ hiện hành
resizeTo(x,y)	Thay đổi kích thước cửa sổ
scrollBy(dx,dy)	Cuộn nội dung sang phải dx, xuống dưới dy pixel
scrollTo(x,y)	Cuộn nội dung trên trang đếnvị trí (x,y)

➢ ***Các phương thức khác***

- ✓ alert("msg")
- ✓ confirm("msg")
- ✓ prompt("msg","init")
- ✓ setTimeout(func,millisec)
- ✓ clearTimeout(ID)
- ✓ setInterval(func,millisec)...

179

Đối tượng trong JavaScript - location

➢ ***Các thuộc tính***

- **hostname**: trả về hostname của URL hiện tại.
- **href**: trả về đường dẫn đầy đủ đến vị trí của tài liệu hiện tại.
- **pathname**: trả về đường dẫn tương đối đến URL hiện tại.
- **port**: trả về cổng nhận thông tin của URL hiện tại, mặc định là cổng 80.

180

Đối tượng trong JavaScript - location

➢ ***Các phương thức***

- **assign**(URL): mở trang mới URL.
- **reload()**: mở lại trang hiện tại.
- **replace**(URL): thay thế trang hiện tại bằng trang mới URL

Ví dụ:

```
location.assign("https://www.google.com.vn");
```

181

Đối tượng trong JavaScript - history

➢ ***Các phương thức***

- **back()**: quay trở lại trang vừa mới được ghé thăm ngay trước khi mở trang hiện tại.
- **forward()**: quay lại trang được mở ngay sau trang hiện tại.
- **go**(number): điều khiển trình duyệt mở trang thứ *number* ngay trước trang hiện tại.

Ví dụ:

```
history.go(-2);          // quay lại trước đó 2 trang
```

182

Đối tượng trong JavaScript - document

❖ **Một số phương thức chính của đối tượng document**

➢ **getElementById**(idValue) : lấy ra element với thuộc tính ***id*** có giá trị là idValue

➢ **getElementsByName**(name): lấy danh sách các element (mảng các element) với thuộc tính ***name*** có giá trị là name

➢ **getElementsByTagName**(tagName) : lấy danh sách các element (mảng các element) với ***thẻ*** có tên là tagName

183

Đối tượng trong JavaScript - document

❖ **Thay đổi nội dung của element**

➢ **inner :** là nội dung chứa bên trong của đối tượng

- **innerHTML :** lấy nội dung text và tag HTML bên trong đối tượng (element).
- **innerText :** chỉ lấy nội dung text bên trong đối tượng.

➢ **outer :** là phần inner và bản thân đối tượng

- **outerHTML :** lấy nội dung text và tag HTML bên trong và của cả đối tượng.
- **outerText :** chỉ lấy nội dung text của đối tượng

184

Đối tượng do người dùng định nghĩa

```
var person = {
  name:"James",
  age: 30,
  address:{
    number: 1,
    street: "Trần Phú"
  },
  intro: function(){
    return " là tổng giám đốc công ty ABC";
  },
  person: function(){
    return;
  }
}
```

185

Đối tượng do người dùng định nghĩa

➢ *Khai báo*

```
function Person(name, job){
  this.name = name;
  this.job = job;
  this.intro = function(){
    return "Tôi tên là " + name + ", tôi là " + job;
  }
}
```

➢ *Sử dụng*

```
var p = new Person("Sơn","sinh viên");
    p.job = "giảng viên";
```

186

Đối tượng do người dùng định nghĩa

➢ *Kế thừa*

```
function Student(name, className){
  Person.call(this, name, "sinh viên");
  this.className = className;
}
```

➢ *Bổ sung thuộc tính hoặc phương thức*

```
Student.prototype.age = 23;
Student.prototype.greeting = function(){
  return "Tôi tên là " + this.name + ", tôi là sinh viên";
}
```

187

Đối tượng do người dùng định nghĩa

```
String.prototype.reverse = function() {
  charArray    = this.split("");
  reverseArray = charArray.reverse();
  reverseText  = reverseArray.join("");
  return reverseText;
}

Array.prototype.rsort = function() {
  return this.sort().reverse();
}
```

188

Xử lý sự kiện

❖ Xử lý sự kiện cho các thẻ HTML

<tag eventHandler = "lệnh javaScript">

Ví dụ:

```
<input type="button" name="Button1" value="OK"
    onclick="window.open('mydoc.html');">
```

189

Xử lý sự kiện

❖ Xử lý sự kiện dạng function

```
<head>
   <script type="text/javascript">
      function GreetingMessage(){
      window.alert("Welcome to my world");
   }
   </script>
</head>
<body onload="GreetingMessage()">
</body>
```

190

Xử lý sự kiện

❖ Xử lý sự kiện dạng thuộc tính

```
object.eventHandler = function_name;
```

```
<body>
   <input type="button" id="bt1" value="OK"/>

   <script type="text/javascript">
     function GreetingMessage(){
        window.alert("Welcome to my world");
     }

     bt = document.getElementById("bt1");
     bt.onclick = GreetingMessage();
   </script>
</body>
```

191

Xử lý sự kiện

❖ Một số sự kiện

➢ *Sự kiện trên chuột*

Sự kiện	Miêu tả
onclick	Thực hiện khi click chuột
ondblclick	Thực hiện khi double click
onmousedown	Thực hiện khi nhấn chuột
onmouseup	Thực hiện khi chuột được thả (Sau khi nhấn)
onmousemove	Thực hiện khi di chuyển chuột
onmouseover	Khi di chuyển chuột lên đối tượng
onmouseout	Khi di chuyển chuột ra khỏi đối tượng

192

Xử lý sự kiện

➢ *Sự kiện trên bàn phím*

Sự kiện	Miêu tả
onkeydown	Thực hiện khi phím được nhấn
onkeyup	Thực hiện khi phím được thả
onkeypress	Thực hiện khi phím được nhấn và thả

193

Xử lý sự kiện

➢ *Sự kiện trên form*

Sự kiện	Miêu tả
onblur	Thực hiện khi phần tử form không được focus
onchange	Thực hiện khi phần tử form được thay đổi (<input>, <select>, và <textarea>
onfocus	Thực hiện khi phần tử form được focus (<label>, <input>, <select>, textarea>, và <button>)
onreset	Khi form được reset
onsubmit	Khi form được submit
onselect	Khi người dùng chọn một đoạn văn bản (<input> và <textarea>)

194

jQuery

❖ Thư viện javascript miễn phí dùng để hỗ trợ cho người lập trình có thể viết JavaScript nhanh hơn, dễ dàng hơn, ngắn gọn hơn.

❖ jQuery đơn giản hóa việc quản lý HTML/DOM, điều khiển sự kiện, chuyển động, CSS.

195

jQuery

❖ **Khai báo**

```
<script type="text/javascript"
 src="js/jquery-3.4.1.min.js"></script>
```

hoặc sử dụng online từ máy chủ CDN của Google

```
<script type="text/javascript"
src="http://code.jquery.com/jquery-3.2.1.min.js">
</script>
```

196

jQuery

❖ Cú pháp

```
$(đối_tượng).điều_khiển();
```

$(this)	Chọn đối tượng hiện tại
$(document)	Chọn đối tượng toàn bộ tài liệu
$(window)	Chọn đối tượng về cửa sổ hiện hành
$('p')	Chọn tất cả thẻ <p>
$('.myClass')	Chọn tất cả thẻ HTML có class="myClass"
$('.myClass').eq(i)	Chọn thẻ HTML thứ i của nhóm *myClass*
$('#idValue')	Chọn thẻ HTML có id="idValue"

197

jQuery – Điều khiển nội dung

❖ Lấy nội dung của đối tượng HTML

```
$(đối_tượng).html();
$(đối_tượng).text();
```

❖ Thiết lập nội dung mới cho đối tượng HTML

```
$(đối_tượng).html('nội dung mới');
$(đối_tượng).text('nội dung mới');
```

Ví dụ:

```
$('#myId').html('Một nội dung mới');
```

198

jQuery – Điều khiển nội dung

❖ Lấy nội dung nhập từ bàn phím của thẻ <input>

```
$(đối_tượng).val();
$(đối_tượng).prop('value');
```

☞ ***Chú ý***: Không dùng .prop() để lấy giá trị của các thuộc tính khác trong phần tử HTML

❖ Thiết lập nội dung hiển thị của thẻ <input>

```
$(đối_tượng).prop('value','giá trị mới');
$(đối_tượng).val('giá trị mới');
```

199

jQuery – Điều khiển nội dung

❖ Thêm nội dung (thêm phần tử HTML)

- `append()` thêm nội dung vào cuối trong phần tử chọn
- `prepend()` thêm nội dung vào phần đầu trong phần tử chọn
- `after()` thêm nội dung vào phía sau phần tử chọn
- `before()` thêm nội dung vào phía trước phần tử chọn

200

jQuery – Điều khiển nội dung

Ví dụ:

```
$('#myId').append('<p>Hello!</p>');
```

201

jQuery – Điều khiển nội dung

❖ Lựa chọn phần tử HTML

- `parent()` lấy phần tử cha trực tiếp của phần tử.
- `parents()` lấy phần tử các phần tử cha (kể cả không trực tiếp)
- `children()` lấy các phần tử con
- `siblings()` các phần tử ngang hàng (anh em)
- `next()` phần tử ngang hàng tiếp theo
- `nextAll()` tất cả các phần tử ngang hàng tiếp theo
- `prev()` phần tử ngang hàng trước
- `prevAll()` tất cả các phần tử ngang hàng phía trước
- `eq(index)` phần tử có thứ tự index trong tập hợp chọn được
- `find(đối_tượng)` tìm phần tử trong các phần tử con, cháu ...

202

jQuery – Điều khiển nội dung

❖ Xóa phần tử HTML

- `empty()`: loại bỏ tất cả các phần tử con của phần tử được chọn.
- `remove()`: loại bỏ các phần tử được chọn.
- `detach()`: tương tự như remove() nhưng toàn bộ dữ liệu jQuery liên kết vẫn được giữ nguyên. Điều này có ích nếu muốn sử dụng lại phần tử đó như sẽ chèn vào vị trí khác.

Ví dụ:

```
$('p').empty();
```

203

jQuery – Điều khiển thuộc tính

❖ Lấy giá trị thuộc tính

```
$(đối_tượng).attr('thuộc tính');
```

❖ Thiết lập giá trị mới cho thuộc tính

```
$(đối_tượng).attr('thuộc tính','giá trị mới');
```

❖ Kiểm tra thuộc tính tồn tại

```
if ($(đối_tượng)[0].hasAttribute('thuộc tính'))
```

❖ Loại bỏ thuộc tính

```
$(đối_tượng).removeAttr('thuộc tính');
```

☞ ***Chú ý***: *Không sử dụng .attr() để lấy/thiết lập giá trị thuộc tính* ***value*** *(thẻ <input>)*

204

jQuery – điều khiển CSS

❖ **Lấy giá trị thuộc tính**

```
$(đối_tượng).css('thuộc tính CSS');
```

❖ **Thiết lập giá trị mới cho thuộc tính CSS**

```
$(đối_tượng).css('thuộc tính CSS',
                 'giá trị mới');
$(đối_tượng).css({'ttCSS_1':'value_1',
                  'ttCSS_2':'value_2',…});
```

Ví dụ:

```
$('p').css('font-family','Arial');
```

205

jQuery – Điều khiển sự kiện

❖ **Gán sự kiện**

```
$(đối_tượng).tên_sự_kiện(hàm_điều_khiển);
```

Ví dụ:

Code HTML

```
<input type="button" value="Click!" id="btClick" />
```

Code jQuery

```
$(document).ready(function(){ // Đăng ký sự kiện
   $('#btClick').click(function(){
      alert('Hello World!');
   });
});
```

206

jQuery – Điều khiển sự kiện

❖ **Hủy sự kiện**

```
$(đối_tượng).unbind('tên_sự_kiện');
$(đối_tượng).unbind(); // hủy toàn bộ sự kiện
```

207

jQuery – Hiệu ứng ẩn/hiện phần tử

❖ **Điều khiển**

show	fadeIn	slideUp
hide	fadeOut	slideDown
toggle	fadeTo	slideToggle

❖ **Cú pháp**

```
$(đối_tượng).điều_khiển(thời gian[,hàm điều khiển]);
```

Ví dụ:

```
$('#exID').fadeTo('slow', 0.7);
```

208

jQuery – API

❖ **Chuyển động, thay đổi hình dạng đối tượng**

```
$(đối_tượng).stop(true).animate(
{
    danh_sách_thuộc_tính
}, thời_gian [, điều_khiển]);
```

```
$('#exID').stop(true).animate(
{
     height :'500px',
     width  :'700px'
},3000 , function(){ alert('Đã xong');});
```

209

HTML5 & CSS3

210

HTML5 – API

❖ **Geolocation**: xác định vị trí

```
<script>
  var x = document.getElementById("demo");
  function getLocation() {
    if (navigator.geolocation) {
      navigator.geolocation.getCurrentPosition(showPosition);
    } else {
      x.innerHTML = "Không hỗ trợ tìm vị trí";
    }
  }
  function showPosition(position) {
    x.innerHTML = "Latitude: " + position.coords.latitude +
            "<br>Longitude: " + position.coords.longitude;
  }
</script>
```

211

HTML5 – API

❖ **Drag/Drop**: di chuyển đối tượng HTML trên trình duyệt

```
<script>
function allowDrop(ev) {
    ev.preventDefault();
}

function drag(ev) {
    ev.dataTransfer.setData("text", ev.target.id);
}

function drop(ev) {
    ev.preventDefault();
    var data = ev.dataTransfer.getData("text");
    ev.target.appendChild(document.getElementById(data));
}
</script>
```

212

HTML5 – API

❖ **Drag/Drop**: di chuyển đối tượng HTML trên trình duyệt

```
<body>
  <div id="div1" ondrop="drop(event)"
                 ondragover="allowDrop(event)">
  </div>
  <img id="drag1" src="img_logo.gif"
    draggable="true" ondragstart="drag(event)"
    width="336" height="69">
</body>
```

213

HTML5 – API

❖ **Session Storage**: dữ liệu được lưu trữ tạm thời chỉ có tác dụng trên một tab duy nhất, không thể chia sẻ với tab khác và sẽ kết thúc khi tab đó bị đóng.

```
sessionStorage.setItem("key", value);
sessionStorage.getItem("key");

// Lưu trữ
sessionStorage.setItem("lastname", "Smith");
// Lấy ra
document.getElementById("result").innerHTML =
sessionStorage.getItem("lastname");
```

214

HTML5 – API

❖ **Local Storage**: giá trị lưu trữ trong local storage không bị mất đi cả khi đóng trình duyệt và có thể được truy xuất bởi các tab khác, thậm chí cả các trang HTML khác.

```
localStorage.setItem("key", value);
localStorage.getItem("key");

// Lưu trữ
localStorage.setItem("lastname", "Smith");
// Lấy ra
document.getElementById("result").innerHTML =
localStorage.getItem("lastname");
```

215

HTML5 – API

❖ **Web Worker**: với mã kịch bản JavaScript khi đang chạy thì trang web sẽ bị khóa lại cho đến khi kịch bản chạy xong thì web được mở và chúng ta có thể thao tác tiếp .

HTML5 web worker là đoạn mã javascript chạy ở chế độ nền, độc lập với kịch bản khác mà không ảnh hưởng tới tương tác cũng như hiệu suất của trang

Web Worker là **không thể truy xuất được thành phần trên DOM**, và cả các đối tượng window, document hay parent. Mã lệnh các công việc cần thực thi cũng phải được **cách ly trong một tập tin script.**

216

HTML5 – API

❖ **Web Worker**

➢ File thực thi *demo_workers.js*

```
var i = 0;

function timedCount() {
    i = i + 1;
    postMessage(i);
    setTimeout("timedCount()",500);
}

timedCount();
```

217

HTML5 – API

➢ Code thực thi *demo_workers.js*

```
var w;
function startWorker() {
    if(typeof(Worker) !== "undefined") {
        if(typeof(w) == "undefined") {
            w = new Worker("demo_workers.js");
        }
        w.onmessage = function(event) {
            document.getElementById("result").innerHTML = event.data;
        };
    } else {
        document.getElementById("result").innerHTML = "No support.";
    }
}

function stopWorker() {
    w.terminate();
    w = undefined;
}
```

218

HTML5 – API

❖ **Server–Sent Events**: Nhận thông tin tự động từ server

➢ Code server *demo_sse.php*

```
<?php
header('Content-Type: text/event-stream');
header('Cache-Control: no-cache');

$time = date('r');
echo "data: The server time is: {$time}\n\n";
flush();
?>
```

219

HTML5 – API

❖ **Server–Sent Events**

➢ Code client nhận dữ liệu tự động từ server

```
if(typeof(EventSource) !== "undefined") {
  var source = new EventSource("demo_sse.php");
  source.onmessage = function(event) {
    document.getElementById("result").innerHTML
                    += event.data + "<br>";
  };
} else {
    // code xử lý không hỗ trợ
}
```

220

HTML5 – API

- **Web SQL Database**: dựa trên đặc tả của CSDL SQLite được xây dựng sẵn trong HTML5. Giống với Local Storage, CSDL này là cục bộ, lâu dài, được quản lí bởi trình duyệt và có thể được sử dụng bởi bất cứ trang HTML nào.
  - **Mở CSDL**

```
db = openDatabase(db_name, db_version, db_desc,
                  db_size);
```

  - **Thực thi câu lệnh SQL**

```
dbTransaction.executeSql('SQL String');
```

221

HTML5

- **Web SQL Database**

```
var db = openDatabase('mydb', '1.0', 'Test DB', 2 * 1024 * 1024);

db.transaction(function (tx) {
   tx.executeSql('CREATE TABLE IF NOT EXISTS LOGS (id unique, log)');
   tx.executeSql('INSERT INTO LOGS (id, log) VALUES (1, "foobar")');
});

db.transaction(function (tx) {
   tx.executeSql('SELECT * FROM LOGS', [], function (tx, results) {
      var len = results.rows.length, i;
      msg = "<p>Found rows: " + len + "</p>";
      document.querySelector('#status').innerHTML +=  msg;

      for (i = 0; i < len; i++) {
         alert(results.rows.item(i).log );
      }
   }, null);
});
```

222

BOOTSTRAP

223

Nội dung

- Giới thiệu về Bootstrap
- Nhúng Bootstrap vào trang web
- Các class CSS Bootstrap

224

Khái niệm

- Bootstrap là 1 framework JavaScript, CSS và HTML miễn phí cho thiết kế giao diện web đáp ứng trên các thiết bị di động.
- Bootstrap được phát triển bởi Mark Otto và Jacob Thornton tại Twitter 8/2011
- Tải gói bootstrap: http://getbootstrap.com/

225

Nhúng Bootstrap vào HTML

```
<!DOCTYPE html>
<html lang="vi">
<head>
  <title>Bootstrap 4 Example</title>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width,
                                 initial-scale=1" />
  <link rel="stylesheet"href="css/bootstrap.min.css" />
  <script src="js/jquery-3.3.1.min.js"></script>
  <script src="js/popper.min.js"></script>
  <script src="js/bootstrap.min.js"></script>
</head>
<body>
...
</body>
</html>
```

226

Sử dụng Bootstrap

- Gọi tên class CSS phù hợp với thẻ HTML mà Bootstrap đã quy định sẵn.
  - *Có thể ghép nhiều giá trị class vào 1 thẻ HTML*

```
<input type="button" class="btn btn-primary btn-lg"
value="Large Button" />
```

227

Cấu trúc lưới Bootstrap

span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1	span 1
span 4			span 4			span 4					
span 4			span 8								
span 6						span 6					
span 12											

- Cú pháp: class="col-x-y"
  - x : Loại thiết bị
  - y : Số cột bị chiếm dụng

228

Cấu trúc lưới Bootstrap

	Extra small <576px	Small ≥576px	Medium ≥768px	Large ≥992px	Extra large ≥1200px
class	.col-	.col-sm-	.col-md-	.col-lg-	.col-xl-

```
<div class="container">
  <h1>Hello World!</h1>
  <div class="row">  <!-- Các đối tượng trong vùng sẽ xếp hàng ngang sát nhau -->
    <div class="col-sm-3 col-md-4 col-lg-6">
      <p>Vùng 1</p>
    </div>
    <div class="col-sm-9 col-md-8 col-lg-6">
      <p>Vùng 2</p>
    </div>
  </div>
</div>
```

229

Cấu trúc lưới Bootstrap

- Có thể sử dụng **offset** để tách rời các vùng

```
<div class="container">
  <div class="row">
    <div class="col-md-3 offset-md-2">
      <p>Cột thứ 1 chiếm 3 vùng, lệch qua phải 2 vùng</p>
    </div>
    <div class="col-md-4 offset-md-1">
      <p>Cột thứ 2 chiếm 4 vùng, lệch phải so với cột 1
        thêm 1 vùng</p>
    </div>
  </div>
</div>
```

230

Icons

❖ **https://getbootstrap.com/docs/4.1/extend/icons/**

- **Sử dụng gói Font Awesome** : xả nén gói và copy file **fontawesome-all.min.js** vào thư mục javascript (js/) của Bootstrap. Thêm khai báo vào file HTML:

```
<script src="js/fontawesome-all.min.js">
</script>
```

- **Sử dụng biểu tượng**

```
<span class="fas fa-tênIcon"></span>
```

```
<span class="fas fa-angle-up"></span>
```

231

Khoảng cách – Kích thước

- **.m{t|r|b|l|x|y}{-sm|md|lg|xl}-{size}** (*margin*)

```
.m-2
.mt-md-5
.mx-auto        (canh giữa theo chiều ngang)
.m-n2           (khoảng cách âm)
```

- **.p{t|r|b|l|x|y}{-sm|md|lg|xl}-{size}** (*padding*)

```
<div class="row mx-md-n5">
  <div class="col px-md-5">
    <div class="p-3 border bg-light">Custom column padding</div>
  </div>
  <div class="col px-md-5">
    <div class="p-3 border bg-light">Custom column padding</div>
  </div>
</div>
```

- **.w-{25|50|75|100|auto}** (*chiều rộng %*)
- **.h-{25|50|75|100|auto}** (*chiều cao %, cần thiết lập trước chiều cao của thành phần chứa đối tượng*)

232

Vị trí

- `.fixed-top` (*cố định đối tượng đầu màn hình*)
- `.fixed-bottom` (*cố định đối tượng cuối màn hình*)
- `.sticky-top` (*đối tượng dừng ở đầu màn hình khi bị cuộn tới*)
- `.float{-sm|md|lg|xl}-{left|right|none}` (*float*)

233

Colors

- `.text-{primary|secondary|success|danger|warning|info|light|dark|body|muted|white|black-50|white-50}` (*Màu chữ*)
- `.bg-{primary|secondary|success|danger|warning|info|light|dark|white|transparent}` (*Màu nền*)
- `.shadow{-sm|lg|none}` (*đổ bóng đối tượng*)

234

Flex

❖ **Bố cục trang web linh động (version 4.x), các đối tượng trong vùng chứa flex được đặt sát nhau.**

- `.d{-sm|md|lg|xl}-flex`

(*Tạo vùng flex chiều rộng tối đa*)

```
<div class="d-flex p-3 bg-secondary text-white">
   <div class="p-2 bg-info w-50">Flex item 1</div>
   <div class="p-2 bg-warning">Flex item 2</div>
   <div class="p-2 bg-primary">Flex item 3</div>
</div>
```

235

Flex (cont.)

- `.d{-sm|md|lg|xl}-inline-flex`

(*Tạo vùng flex chiều rộng vừa đủ nhóm đối tượng*)

```
<div class="d-md-inline-flex p-3 bg-secondary text-white">
   <div class="p-2 bg-info">Flex item 1</div>
   <div class="p-2 bg-warning">Flex item 2</div>
   <div class="p-2 bg-primary">Flex item 3</div>
</div>
```

236

Flex (cont.)

- **.flex{-sm|md|lg|xl}-row-reverse**

  (Các đối tượng trong vùng flex được xếp ngược lại bên phải)

```
<div class="d-flex flex-row-reverse p-3 bg-secondary text-white">
   <div class="p-2 bg-info w-50">Flex item 1</div>
   <div class="p-2 bg-warning">Flex item 2</div>
   <div class="p-2 bg-primary">Flex item 3</div>
</div>
```

237

Flex (cont.)

- **.flex{-sm|md|lg|xl}-column**

  (Các đối tượng trong vùng flex được xếp dọc)

- **.flex{-sm|md|lg|xl}-column-reverse**

  (Các đối tượng trong vùng flex được xếp dọc ngược lại)

```
<div class="d-inline-flex flex-column p-3 bg-secondary text-white">
   <div class="p-2 bg-info w-50">Flex item 1</div>
   <div class="p-2 bg-warning">Flex item 2</div>
   <div class="p-2 bg-primary">Flex item 3</div>
</div>
```

238

Flex (cont.)

- **.flex{-sm|md|lg|xl}-wrap**

  (Cho phép xuống dòng nếu số lượng đối tượng quá nhiều)

```
<div class="d-flex flex-wrap m-5 p-3 bg-light">
   <div class="p-2 border bg-success">Flex item 1</div>
   <div class="p-2 border bg-success">Flex item 2</div>
   ...
   <div class="p-2 border bg-success">Flex item 15</div>
</div>
```

- **.flex{-sm|md|lg|xl}-wrap-reverse**

  (Cho phép xuống dòng và đảo thứ tự đối tượng)

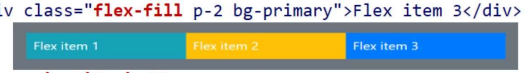
239

Flex (cont.)

- **.flex{-sm|md|lg|xl}-fill**

  (Các vùng flex chiếm toàn bộ bề rộng của đối tượng)

```
<div class="d-flex p-3 bg-secondary text-white">
   <div class="flex-fill p-2 bg-info">Flex item 1</div>
   <div class="flex-fill p-2 bg-warning">Flex item 2</div>
   <div class="flex-fill p-2 bg-primary">Flex item 3</div>
</div>
```



- **.flex{-sm|md|lg|xl}-grow-1**

  (Đối tượng có chiều rộng là phần còn lại của vùng flex)

```
<div class="d-flex p-3 bg-secondary text-white">
   <div class="p-2 bg-info">Flex item 1</div>
   <div class="p-2 bg-warning">Flex item 2</div>
   <div class="flex-grow-1 p-2 bg-primary">Flex item 3</div>
</div>
```

240

Flex (cont.)

- `.order{-sm|md|lg|xl}-{0|1|...|12}`

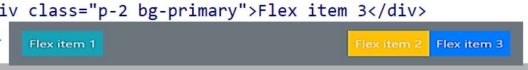
(Đặt thứ tự cho đối tượng)

```
<div class="d-inline-flex p-3 bg-secondary text-white">
    <div class="order-1 p-2 bg-info">Flex item 1</div>
    <div class="order-0 p-2 bg-warning">Flex item 2</div>
    <div class="order-2 p-2 bg-primary">Flex item 3</div>
</div>
```

- `.{mr|ml|mt|mb}-auto`

(Đẩy các đối tượng còn lại qua phải | trái | trên | dưới của vùng)

```
<div class="d-flex p-3 bg-secondary text-white">
    <div class="mr-auto p-2 bg-info">Flex item 1</div>
    <div class="p-2 bg-warning">Flex item 2</div>
    <div class="p-2 bg-primary">Flex item 3</div>
</div>
```



241

Flex (cont.)

- `.justify-content{-sm|md|lg|xl}`
  `-{start|end|center|between|around}`

(Canh lề các đối tượng trong vùng flex theo chiều ngang)

242

Flex (cont.)

- `.align-content{-sm|md|lg|xl}`
  `-{start|end|center|between|around|stretch}`

*(Canh lề **tất cả đối tượng** trong vùng flex theo chiều dọc, thường đi kèm với **flex-wrap**, sử dụng khi có quá nhiều đối tượng trong vùng flex)*

243

Flex (cont.)

- `.align-items{-sm|md|lg|xl}`
  `-{start|end|center|baseline|stretch}`

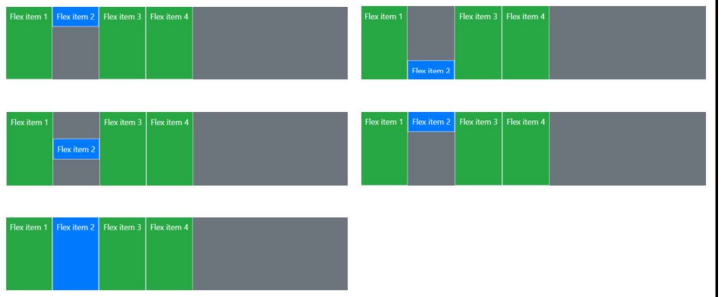
*(Canh lề **dòng đối tượng** trong vùng flex theo chiều dọc)*

244

Flex (cont.)

- `.align-self{-sm|md|lg|xl}`
  `-{start|end|center|baseline|stretch}`

  (*Canh lề **riêng đối tượng** trong vùng flex theo chiều dọc*)



245

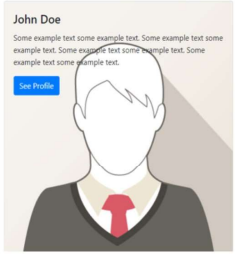
Thẻ (Card)

- `.card` (*Xác định vùng thẻ*)
- `.card-body` (*Vùng nội dung chính của thẻ*)
- `.card-{header|footer}` (*Vùng đầu|cuối thêm vào thẻ*)
- `.card-{title|text|link}` (*Các loại văn bản trong card-body*)
- `.card-img-{top|bottom}` (*thẻ <img> - Vị trí hình trong thẻ, đặt phía ngoài **card-body***)
- `.stretched-link` (*thẻ <a> - Vùng thẻ sẽ click được*)
- `.card-img-overlay` (*Sử dụng chung hoặc thay thế **card-body** để chuyển hình ảnh thành background*)

246

Thẻ (Card) (cont.)

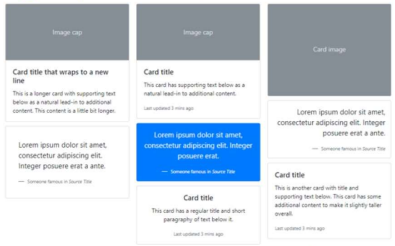
```
<div class="card img-fluid" style="width:500px">
  <img class="card-img-top"
       src="img_avatar1.png" alt="Card image"
       style="width:100%">
  <div class="card-img-overlay">
    <h4 class="card-title">John Doe</h4>
    <p class="card-text">Some example text
some example text. Some example text some
example text. Some example text some example
text. Some example text some example
text.</p>
    <a href="#" class="btn btn-primary stretched-link">See Profile</a>
  </div>
</div>
```



247

Nhóm các thẻ (card)

- `.card-column` (*Nhóm các thẻ được chia **chiều rộng bằng nhau**, số lượng thẻ trên 1 dòng được chia đều **tùy thuộc vào tổng số thẻ**; các thẻ sẽ được nổi sát nhau tương tự float nhưng được tách ra.*)

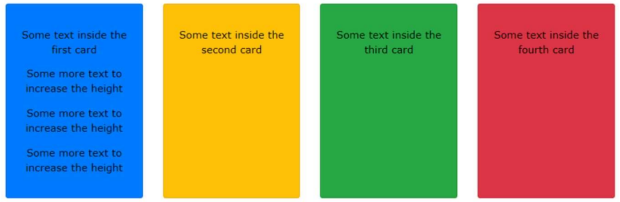


☞ ***Chú ý***: card-column như trên chỉ được hiển thị với màn hình ≥ **sm**

248

Nhóm các thẻ (card)

- `.card-deck` (*Nhóm các thẻ đặt sát nhau nhưng **tách ra** được chia **chiều rộng và chiều cao bằng nhau** tùy vào nội dung*)

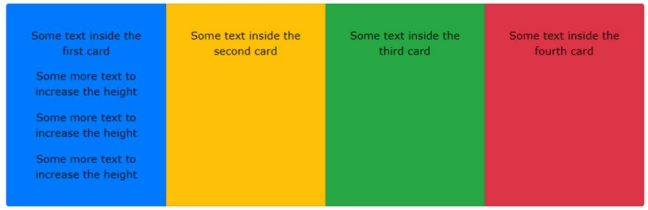


☞ ***Chú ý***: card-deck như trên chỉ được hiển thị với màn hình ≥ **sm**

249

Nhóm các thẻ (card)

- `.card-group` (*Nhóm các thẻ đặt sát nhau được chia **chiều rộng và chiều cao bằng nhau** tùy vào nội dung*)



☞ ***Chú ý***: card-group như trên chỉ được hiển thị với màn hình ≥ **sm**

250

Vùng hiển thị

- `.container{-fluid}` (*Vùng hiển thị giữa màn hình*)
- `.jumbotron{-fluid}` (*Vùng hiển thị nổi*)
- `.alert{-*}` (*Vùng hiển thị 1 dòng*)

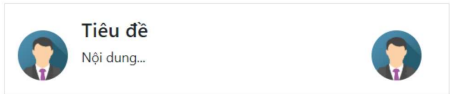
```
<div class="alert alert-primary alert-dismissible fade show">
   <button type="button" class="close" data-dismiss="alert">
       &times;
   </button>
   <strong>Primary!</strong> Indicates an important action.
</div>
```

251

Bố cục các đối tượng media

- `.media` (*Xác định vùng hiển thị có media – hình, video…*)
- `.media-body` (*Vùng text nội dung*)

```
<div class="media border p-3">
   <img src="image_1" alt="h1" class="mr-3 mt-3 rounded-circle"
                      style="width:60px;">
   <div class="media-body">
      <h4>Tiêu đề</h4>
      <p>Nội dung...</p>
   </div>
   <img src="image_2" alt="h2" class="mr-3 mt-3 rounded-circle"
                      style="width:60px;">
</div>
```



252

Viền (Border)

- `.border{-top|right|bottom|left}` (*Kẻ viền*)
- `.border{-top|right|bottom|left}-0` (*Xóa viền*)
- `.border-{primary|secondary|...}` (*Màu viền*)
- `.round{-sm|lg|circle|top|right|bottom|left|0}` (*Bo tròn góc*)

253

Text

- `.display-1|.display-2|.display-3|.display-4|.lead` (*Kích thước chữ*)
- `.font-*` (*Điều chỉnh font chữ*)
  - `.font-weight-{lighter|light|bold|bolder|normal}`
  - `.font-italic`
- `.text-*` (*Điều chỉnh văn bản*)
  - `.text{-sm|md|lg|xl}-{left|right|center|justify}`
  - `.text-decoration-none`
  - `.text-{lowercase|uppercase|capitalize}`
  - `.text-reset`
- `.list-*` (*Điều chỉnh danh sách – thẻ <ul> hoặc *)
  - `.list-unstyled`
  - `.list-inline` (dùng chung với `.list-inline-item` trong <li>)

254

Table

- `.table` (*thẻ <table> - Tạo bảng cơ bản*)
- `.table-striped` (*thẻ <table> - Bảng phân biệt dòng*)
- `.table-bordered` (*thẻ <table> - Bảng có viền*)
- `.table-borderless` (*thẻ <table> - Bảng không viền*)
- `.table-hover` (*thẻ <table> - Bảng có hover từng dòng*)
- `.table-*` (*thẻ <table>, <tr>, <td> - Bảng có màu **)
  - `.table-primary` (bảng được phủ màu primary)
- `.table-responsive{-sm|md|lg|xl}` (*thẻ chứa bảng - Bảng hiển thị thanh trượt tùy màn hình*)

255

Hình ảnh - <img>

- `.rounded` (*Hình có viền tròn*)
- `.rounded-circle` (*Hình cắt hiển trị tròn*)
- `.img-thumbnail` (*Hình hiển thị dạng thumbnail, có viền*)
- `.float{-sm|md|lg|xl}-{left|right}` (*Vị trí hình bên trái | phải*)
- `.mx-auto d-block` (*Vị trí hình chính giữa*)
- `.img-fluid` (*Hình ảnh co giãn tùy kích thước màn hình*)

256

Nút bấm (button)

❖ **Nên sử dụng với thẻ <input>, <a> và <button>**

- `.btn` (*Xác định nút bấm, nếu thẻ <a> cần có thêm* **role="button"**)
- `.btn-{primary|secondary|...}` (*Màu nút bấm*)
- `.btn-outline-{primary|secondary|...}` (*Nút bấm có viền và hover màu*)
- `.btn-block` (*Nút bấm full vùng chứa*)
- `.btn-{sm|lg}` (*Kích thước nút bấm*)

257

Nhóm nút bấm (button group)

- `.btn-group` (*Xác định nhóm nút bấm*)
  - ☞ ***Chú ý***: cần thêm **role="group"** và **aria-label**="*tên nhóm*" vào thẻ)
- `.btn-toolbar` (*Xác định toolbar nút bấm và các đối tượng khác*)
  - ☞ ***Chú ý***: cần thêm **role="toolbar"** và **aria-label**="*tên toolbar*" vào thẻ)
- `.btn-group{-sm|lg}` (*Kích thước nhóm nút bấm*)
- `.btn-group-vertical` (*Nhóm nút bấm chiều dọc*)

258

Nhóm danh sách (list group)

- `.list-group` (*Xác định nhóm danh sách dạng cột*)
- `.list-group-flush` (*Danh sách đơn giản*)
- `.list-group-horizontal{-sm|md|lg|xl}` (*Danh sách theo hàng ngang*)
- ✓ `.list-group-item` (*Xác định phần tử của danh sách*)
- ✓ `.active` (*Phần tử danh sách được chọn*)
- ✓ `.disabled` (*Phần tử danh sách bị vô hiệu*)
- ✓ `.list-group-item-{*}` (*Phần tử danh sách có màu **)
- ✓ `.list-group-item-action` (*Phần tử danh sách có hover*)

259

Thẻ đánh dấu (badge)

- `.badge` (*Xác định thẻ đánh dấu*)
- `.badge-{primary|secondary|...}` (*Màu thẻ*)
- `.badge-pill` (*Bo tròn thẻ*)

```
<h1>
  Example badge
  <span class="badge badge-success badge-pill">New</span>
</h1>
```

Example badge New

260

Vùng phân trang (Pagination)

- **.pagination** (*Xác định vùng phân trang*)
- **.pagination-{sm|lg}** (*Kích thước vùng phân trang*)
- **.page-item** (*Xác định 1 đơn vị trang*)
- **.page-link** (*Link của 1 đơn vị trang*)

```
<ul class="pagination pagination-lg">
   <li class="page-item disabled">
      <a class="page-link" href="#">Previous</a>
   </li>
   <li class="page-item active">
      <a class="page-link" href="#">1</a>
   </li>
   <li class="page-item"><a class="page-link" href="#">2</a></li>
   <li class="page-item"><a class="page-link" href="#">3</a></li>
   <li class="page-item"><a class="page-link" href="#">Next</a></li>
</ul>
```

Previous 1 2 3 Next

261

Dropdown

- **.dropdown** (*tạo vùng dropdown, trong vùng có nút bấm kích hoạt* **.dropdown-toggle** *và menu* **.dropdown-menu**, *trong menu có các* **.dropdown-item**)

```
<div class="dropdown">
  <button type="button" class="btn btn-primary dropdown-toggle"
                        data-toggle="dropdown">
    Dropdown button
  </button>
  <div class="dropdown-menu">
    <a class="dropdown-item" href="#">Link 1</a>
    <a class="dropdown-item" href="#">Link 2</a>
    <a class="dropdown-item" href="#">Link 3</a>
  </div>
</div>
```

Dropdown button
Link 1
Link 2
Link 3

262

Dropdown (cont.)

- **.dropdown-divider** (*Phân vùng dropdown-menu bằng đường thẳng*)
- **.dropdown-header** (*Phân vùng dropdown-menu bằng dòng text header*)
- **.dropdown-menu-right** (*Menu dropdown sẽ nằm bên phải nếu vùng dropdown quá dài*)
- **.dropup|dropright|dropleft** (*Vị trí hiển thị menu dropdown* ***nếu có không gian***)

263

Ẩn/hiện vùng nội dung (Collapse)

```
<button data-toggle="collapse" data-target="#demo">
  Collapsible
</button>
<div id="demo" class="collapse">
  Vùng nội dung sẽ ẩn/hiện (mặc định ban đầu là ẩn)
</div>
```

☞ ***Chú ý***: Nếu là thẻ <a> thì thay thế **data-target** bằng **href**

```
<a data-toggle="collapse" href="#demo">Collapsible</a>
```

264

Ẩn/hiện vùng nội dung (Collapse)

☞ ***Chú ý***: Nếu muốn nhiều vùng ẩn hiện xen kẽ thì sử dụng thuộc tính **data-parent**

```
<div id="parent">
  <div class="card">
    <div class="card-header">
      <a class="card-link" data-toggle="collapse" href="#collapseOne">Vùng hiển thị #1</a>
    </div>
    <div id="collapseOne" class="collapse show" data-parent="#parent">
      <div class="card-body">Khi vùng 1 hiện thì các vùng khác ẩn</div>
    </div>
  </div>
  <div class="card">
    <div class="card-header">
      <a class="collapsed card-link" data-toggle="collapse" href="#collapseTwo">Vùng hiển thị #</a>
    </div>
    <div id="collapseTwo" class="collapse" data-parent="#parent">
      <div class="card-body">Khi vùng 2 hiện thì các vùng khác ẩn</div>
    </div>
  </div>
</div>
```

265

Menu (nav)

- **.nav** *(thẻ <ul> - Tạo menu)*
- **.nav-item** *(thẻ <li> - Tạo phần tử menu)*
- **.nav-link** *(thẻ <a> - Liên kết trong mỗi .nav-item)*

```
<ul class="nav">
  <li class="nav-item">
    <a class="nav-link" href="#">Link 1</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" href="#">Link 2</a>
  </li>
  <li class="nav-item">
    <a class="nav-link disabled" href="#">Disabled</a>
  </li>
</ul>
```

Link 1 Link 2 Disabled

266

Menu (nav) (cont.)

- **.nav-tabs** *(Chuyển menu dạng tab)*

```
<ul class="nav nav-tabs">
```

Active | Link | Link | Disabled

- **.nav-pills** *(Chuyển menu dạng pill)*

```
<ul class="nav nav-pills">
```

Active Link Link Disabled

- **.nav-justified** *(Các phần tử của menu có độ rộng bằng nhau)*

```
<ul class="nav nav-pills nav-justified">...</ul>
```

267

Menu (nav) (cont.)

```
<!-- Menu -->
<ul class="nav nav-tabs">
  <li class="nav-item">
    <a class="nav-link active" data-toggle="tab" href="#menu1">Menu 1</a>
  </li>
  <li class="nav-item">
    <a class="nav-link" data-toggle="tab" href="#menu2">Menu 2</a>
  </li>
</ul>

<!-- Vùng hiển thị nội dung -->
<div class="tab-content">
  <div class="tab-pane container active" id="menu1">...</div>
  <div class="tab-pane container fade" id="menu2">...</div>
</div>
```

268

Vùng menu (navbar)

- **.navbar** (*Xác định vùng menu ngang*)
- **.navbar-expand-{xl|lg|md|sm}**
  (*Vùng menu sẽ chuyển thành dọc tương ứng kích thước màn hình*)
- **.navbar-text** (*Tạo đoạn text trong vùng*)
- **.navbar-brand** (*Tạo nhãn cho vùng*)
- **.navbar-nav** (*thẻ <ul> - Tạo menu trong vùng*)
- **.nav-item** (*thẻ <li> - Tạo phần tử menu*)
- **.nav-link** (*thẻ <a> - Liên kết trong mỗi .nav-item*)

269

Vùng menu (navbar) (cont.)

- **.navbar-toggler** (*Xác định menu sẽ bị thu nhỏ*)

```
<nav class="navbar navbar-expand-md bg-dark navbar-dark">
  <a class="navbar-brand" href="#">Navbar</a>

  <!-- Button ẩn hiện menu khi màn hình nhỏ -->
  <button class="navbar-toggler" data-toggle="collapse"
                                 data-target="#collapsibleNavbar">
    <span class="navbar-toggler-icon"></span>
  </button>

  <!-- Menu -->
  <div class="collapse navbar-collapse" id="collapsibleNavbar">
    <ul class="navbar-nav">
    ...
    </ul>
  </div>
</nav>
```

270

Vùng menu (navbar) (cont.)

```
<nav class="navbar navbar-expand-md bg-dark navbar-dark d-flex"
     id="parent">
  <div class="navbar-toggler text-center w-100 border-0 p-0">
    <!-- Nút bấm 1 bên trái (float-left) đại diện cho menu 1 -->
    <button class="navbar-toggler float-left" data-toggle="collapse"
            data-target="#collapsibleNavbar_1">
      <span class="navbar-toggler-icon"></span>
    </button>
    <!-- Tiêu đề của vùng-->
    <a class="navbar-text text-decoration-none text-white"
       href="#">Khoa CNTT</a>
    <!-- Nút bấm 2 bên phải (float-right) đại diện cho menu 2 -->
    <button class="navbar-toggler float-right" data-toggle="collapse"
            data-target="#collapsibleNavbar_2">
      <span class="navbar-toggler-icon"></span>
    </button>
  </div>
```

271

Vùng menu (navbar) (cont.)

```
<!- Menu 1 -->
<div class="collapse navbar-collapse" id="collapsibleNavbar_1"
     data-parent="#parent">
  <ul class="navbar-nav">
    <li class="nav-item"><a class="nav-link" href="#">Trang chủ</a></li>
    <li class="nav-item dropdown"> <!-- menu ghép -->
      <a class="nav-link dropdown-toggle" href="#" role="button"
         data-toggle="dropdown">
         Menu ghép <!-- Nhãn menu hiển thị trên menu chính  -->
      </a>
      <div class="dropdown-menu"> <!-- các menu con -->
        <a class="dropdown-item" href="#">Mục 1</a>
        <div class="dropdown-divider"></div> <!-- Đường phân cách -->
        <a class="dropdown-item" href="#">Mục cuối cùng</a>
      </div>
    </li>
    <li class="nav-item"> <!-- menu bị vô hiệu hóa -->
      <a class="nav-link disabled" href="#">Mục vô hiệu</a>
    </li>
  </ul>
</div> <!-- kết thúc Menu 1 -->
```

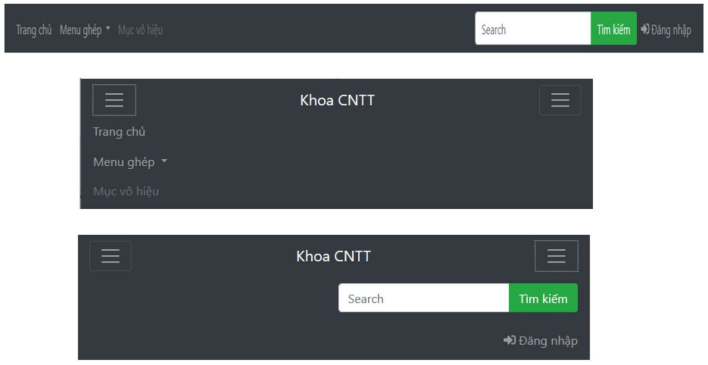
272

Vùng menu (navbar) (cont.)

```
<!-- Menu 2 -->
   <div class="collapse navbar-collapse justify-content-end text-right"
        id="collapsibleNavbar_2" data-parent="#parent">
      <form class="form-inline my-3 my-md-0 justify-content-end">
         <div class="input-group">
            <input type="text" class="form-control" placeholder="Search" />
            <div class="input-group-append">
               <button class="btn btn-success" type="submit">Tìm kiếm</button>
            </div>
         </div>
      </form>
      <ul class="navbar-nav">
         <li class="nav-item">
            <a class="nav-link" href="#">
               <span class="fas fa-sign-in-alt mr-1"></span>Đăng nhập
            </a>
         </li>
      </ul>
   </div> <!-- kết thúc Menu 2 -->
</nav>
```

273

Vùng menu (navbar) (cont.)



274

Form

- **.form-control** *(thẻ <input>, <textarea>, <select> - đối tượng sẽ có chiều rộng tối đa)*
- **.form-inline** *(thẻ <form> - tạo form hàng ngang đối với màn hình sm (≥ 576px) trở lên)*

275

Input

- **.custom-control custom-{checkbox|switch|radio}**
  (Tạo vùng hiển thị checkbox, radio)

```
<div class="custom-control custom-checkbox">
   <input type="checkbox" class="custom-control-input" id="customCheck">
   <label class="custom-control-label" for="customCheck">
      Custom checkbox
   </label>
</div>

<div class="custom-control custom-switch">
   <input type="checkbox" class="custom-control-input" id="switch1">
   <label class="custom-control-label" for="switch1">Toggle me</label>
</div>
```

Custom checkbox

Toggle me

276

Input (cont.)

- **.custom-select{-sm|lg}** *(Tạo đa lựa chọn select)*

```
<select name="cars" class="custom-select">
  <option>Toyota</option>
  <option>Honda</option>
</select>
```

- **.custom-range** *(Tạo lựa chọn range)*

```
<input type="range" class="custom-range" />
```

- **.custom-file** *(Tạo ô nhập file)*

```
<div class="custom-file">
  <input type="file" class="custom-file-input" id="customFile">
  <label class="custom-file-label" for="customFile">
    Choose file
  </label>
</div>
```

277

Input group

- **.input-group** *(Xác định vùng nhóm nhập dữ liệu)*
- **.input-group-{sm|lg}** *(Kích thước vùng)*
- **.input-group-{prepend|append}**
  (Vị trí vùng nhập dữ liệu ở đầu|cuối trong nhóm)
- **.input-group-text** *(Vùng dành cho text, checkbox, radio)*

278

Input group (cont.)

```
<div class="input-group input-group-lg">
  <div class="input-group-prepend">
    <span class="input-group-text">Text</span>
    <div class="input-group-text">
      <input type="radio">
    </div>
    <button class="btn btn-primary" type="button">Button</button>
  </div>
  <input type="text" class="form-control">
  <div class="input-group-append">
    <button type="button" class="btn btn-outline-success dropdown-toggle"
                          data-toggle="dropdown">
      Dropdown button
    </button>
    <div class="dropdown-menu">
      <a class="dropdown-item" href="#">Link 1</a>
      <a class="dropdown-item" href="#">Link 2</a>
    </div>
  </div>
</div>
```

Text | Button | Dropdown button

279

Slide ảnh (Carousel)

- **.carousel slide** *(Xác định vùng slide ảnh, có thêm* **data-ride="carousel"** *sẽ kích hoạt tự động chạy ảnh)*
- **.carousel-indicators** (Nút chọn ảnh)
- **.carousel-inner** (Vùng nhóm ảnh)
- **.carousel-item** (Vùng ảnh trong nhóm ảnh)
- **.carousel-caption** (Vùng text trong vùng ảnh)
- **.carousel-control-prev** (Nút chuyển ảnh trước đó)
- **.carousel-control-next** (Nút chuyển ảnh tiếp theo)

280

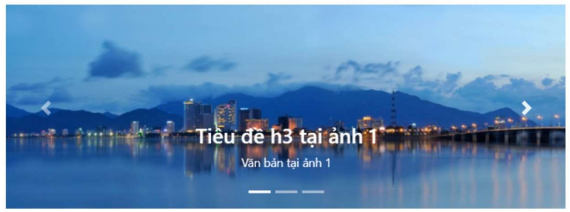
Slide ảnh (Carousel) (cont.)

```
<div id="demo" class="carousel slide" data-ride="carousel">

  <!-- Nút chọn ảnh -->
  <ul class="carousel-indicators">
     <li data-target="#demo" data-slide-to="0" class="active"></li>
     <li data-target="#demo" data-slide-to="1"></li>
  </ul>
  <!-- Silde ảnh (2 ảnh trong slide) -->
  <div class="carousel-inner">
     <div class="carousel-item active">
        <img src="image_1.jpg" alt="Ảnh 1">
        <div class="carousel-caption">
           <h3>Tiêu đề h3 tại ảnh 1</h3>
           <p>Văn bản tại ảnh 1</p>
        </div>
     </div>
     <div class="carousel-item">
        <img src="image_2.jpg" alt="Ảnh 2"> <!-- Ảnh 2 không có văn bản -->
     </div>
  </div>
```

281

Slide ảnh (Carousel) (cont.)

```
    <!-- Điều khiển chuyển ảnh-->
    <a class="carousel-control-prev" href="#demo" data-slide="prev">
       <span class="carousel-control-prev-icon"></span>
    </a>
    <a class="carousel-control-next" href="#demo" data-slide="next">
       <span class="carousel-control-next-icon"></span>
    </a>

 </div>
```



282

Vùng hiển thị nổi (Modal)

- **.modal** (Xác định vùng hiển thị nổi)
- **.modal-dialog** (Xác định vùng tương tác)
- **.modal-{sm|lg|xl}** (Kích thước vùng tương tác)
- **.modal-dialog-center** (Vùng tương tác chính giữa màn hình)
- **.modal-dialog-scrollable** (Tạo thêm scroll)
- **.modal-content** (Tạo vùng nội dung)
- **.modal-header** (Phần đầu nội dung)
- **.modal-body** (Phần thân nội dung)
- **.modal-footer** (Phần chân nội dung)

283

Vùng hiển thị nổi (Modal) (cont.)

```
<!-- Nút bấm hiển thị -->
<button type="button" class="btn btn-primary" data-toggle="modal"
                                              data-target="#myModal">
  Open modal
</button>

<!-- Modal -->
<div class="modal fade" id="myModal">
  <div class="modal-dialog modal-lg modal-dialog-centered
                           modal-dialog-scrollable">
    <div class="modal-content">
```

284

Vùng hiển thị nổi (Modal) (cont.)

```
        <!-- Modal Header -->
        <div class="modal-header">
           <h4 class="modal-title">Modal Heading</h4>
           <button type="button" class="close" data-dismiss="modal">
              &times;
           </button>
        </div>
        <!-- Modal body -->
        <div class="modal-body">
           Modal body..
        </div>
        <!-- Modal footer -->
        <div class="modal-footer">
           <button type="button" class="btn btn-danger" data-dismiss="modal">
              Close
           </button>
        </div>
     </div>   <!-- Kết thúc modal-content -->
  </div>   <!-- Kết thúc modal-dialog -->
</div>   <!-- Kết thúc modal -->
```

285

286