HỌC PHẦN "LẬP TRÌNH WEB VỚI PHP"

(Web programming with PHP)

GVGD: ThS. Trần Mạnh Đông

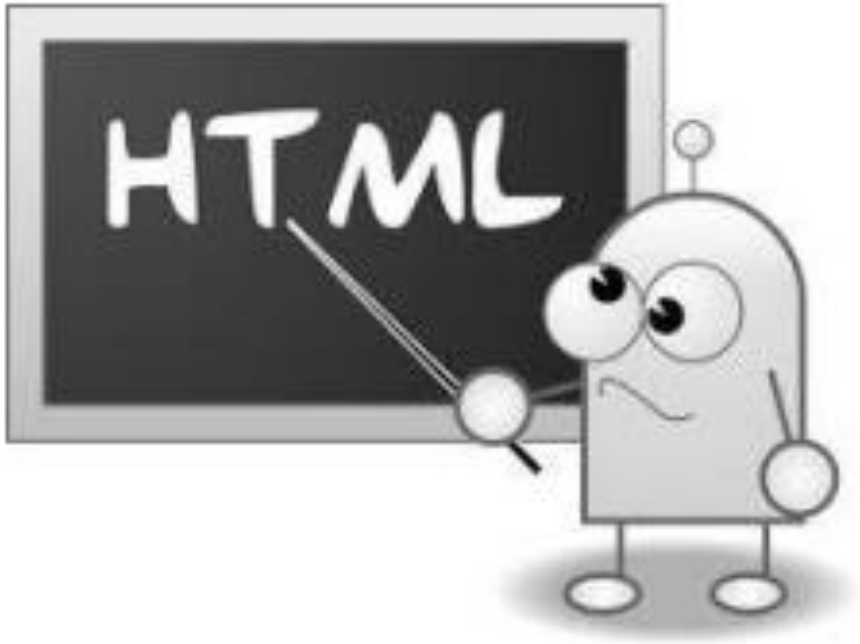
Nội dung

- HTML
- CSS
- JavaScript

1. Nhắc lại về HTML

HTML cơ bản

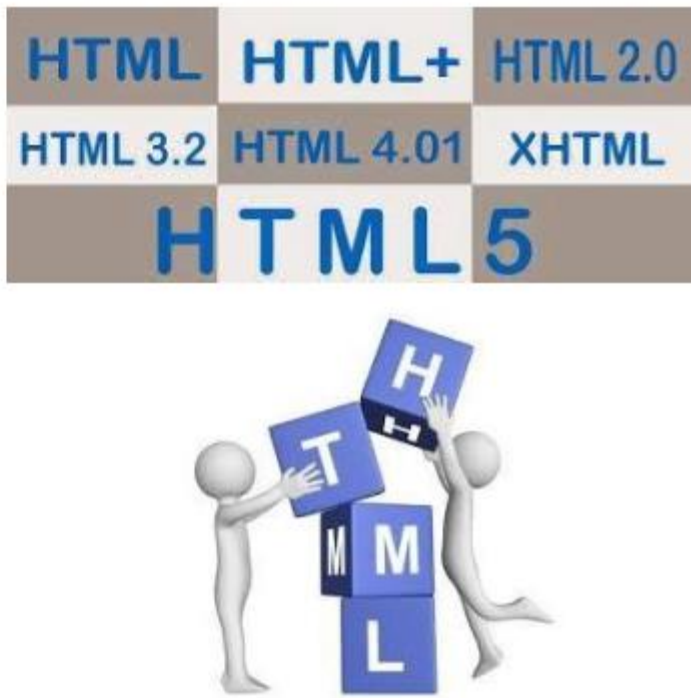
- HTML (HyperText Markup Language – Ngôn ngữ đánh dấu siêu văn bản): Ngôn ngữ để viết các trang web. Do Tim Berner Lee phát minh và được W3C (World Wide Web Consortium) đưa thành chuẩn năm 1994.



HTML cơ bản...

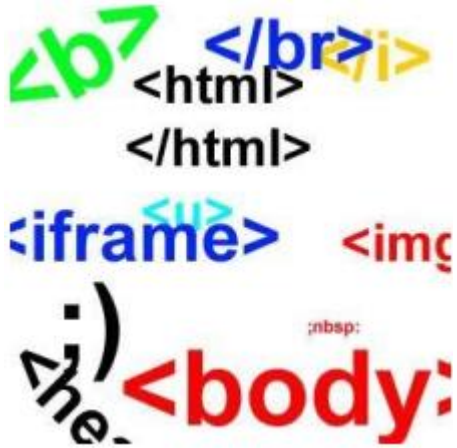
- **Lịch sử HTML**

Version	Year
HTML	1991
HTML+	1993
HTML 2.0	1995
HTML 3.2	1997
HTML 4.01	1999
XHTML	2000
HTML5	2012



HTML cơ bản…

- **Đặc điểm**
  - HTML sử dụng các thẻ (tags) để **định dạng** dữ liệu
  - HTML không phân biệt chữ hoa, chữ thường
  - Các trình duyệt thường không báo lỗi cú pháp HTML. Nếu viết sai cú pháp chỉ dẫn đến kết quả hiển thị không đúng với dự định



HTML cơ bản...

- **Cấu trúc file HTML**

```
<html>
    <body>
        <h1>This is a heading</h1>
        <p>This is a paragraph.</p>
        <p>This is another paragraph.</p>
    </body>
</html>
```

HTML cơ bản...

- **Khai báo doctype giúp trình duyệt hiển thị trang web chính xác**
  - HTML5

```
<!DOCTYPE html>
```

  - HTML 4.01

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

  - XHTML 1.0

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

HTML cơ bản…

- **Thẻ (tag)**
  - Có nhiều thẻ, mỗi thẻ có 1 tên và mang ý nghĩa khác nhau
  - Có 2 loại thẻ: thẻ mở và thẻ đóng

```
<tagname> content </tagname>
```

  - Cách viết thẻ:
    - Thẻ mở: **<tên_thẻ>**
      - Ví dụ: <u>, , <img>…
    - Thẻ đóng tương ứng: **</tên_thẻ>**
      - Ví dụ: </u>,
  - *Lưu ý*: luôn có thẻ mở nhưng có thể không có thẻ đóng tương ứng. Ví dụ: <img>,
, <hr>, <meta>… không có thẻ đóng

HTML cơ bản…

- **Thẻ <html>…</html> cho biết đây là tài liệu có định dạng HTML**

```
<!doctype html>
<html>
        <head>
                <title>Hello World!</title>
        </head>
<body>
  <h1>Hello HTML!</h1>
   <img src="images/logo.png" alt="" />
        <a href="http://google.com">Google</a>

</body>
</html>
```

HTML cơ bản…

- **Thẻ <head>...</head> : chứa một số thông tin của trang**
  - Tiêu đề: <title>…</title>
  - Các thẻ mở rộng
  - Các đường link tới một số file khác
  - Nội dung trong thẻ head không được hiển thị trong cửa sổ trình duyệt
- Thẻ **<body>...</body>**: chứa toàn bộ nội dung của trang và được hiển thị trên cửa sổ trình duyệt

HTML cơ bản...

```
<!doctype html>
<html>
<head>
		<title>Lập trình WEB</title>
		<style>
			h1
			{
				color:red;
			}
		</style>
		<script>
			function onClick()
			{
				alert("Hello from JavaScript");
			}
		</script>
</head>
<body>
  <h1>HTML cơ bản!</h1>
		<a href="http://google.com">Google</a>
  <hr width="95%" size="3px"/>
  <img src="images/logo.png" onclick="onClick()"/>
</body>
</html>
```

HTML cơ bản…

- **Thuộc tính (property) của thẻ**
  - Một thẻ có thể có các thuộc tính nhằm bổ sung tác dụng cho thẻ
  - Mỗi thuộc tính có tên thuộc tính (tên_TT)
  - *Các thuộc tính đặt trong thẻ mở*
  - Viết thẻ có thuộc tính

```
<tên_thẻ tên_TT1="giá_trị1" tên_TT2="giá_trị2"…>
```

  - Ví dụ

```
    <body bgcolor="green" >
        <h1 align="center" >HTML cơ bản</h1>
        <font color="white" face="arial"> Định dạng font chữ</font>
    </body>
</html>
```

HTML cơ bản…

- **Thuộc tính (property) của thẻ…**
  - Có thể thay đổi thứ tự, số lượng các thuộc tính mà không gây ra lỗi cú pháp

```
<body bgcolor="green" >
    <h1 align="center" >HTML cơ bản</h1>
    <font color="white"> Định dạng font chữ 1</font>
    <font color="white" face="arial"> Định dạng font chữ 2</font>
    <font face="arial" color="white"> Định dạng font chữ 3</font>
</body>
```

  - Sự hỗ trợ các thẻ, thuộc tính ở mỗi trình duyệt là khác nhau. Chỉ giống nhau ở các thẻ, thuộc tính cơ bản
  - Thẻ đóng của thẻ có thuộc tính vẫn **viết bình thường** (</tên_thẻ>)

HTML cơ bản...

- **Thuộc tính (property) của thẻ…**
  - Một số thuộc tính
    - **background**: Dùng cho định dạng file hình ảnh làm nền (.gif, .jpg, .bmp).
    - **bgcolor** : Xác lập màu cho nền.
    - **text** : màu chữ.
    - **link** : màu cho liên kết chưa xem.
    - **vlink** : màu cho liên kết đã xem.
    - **alink** : màu cho liên kết đang xem.
    - **leftmargin** : Canh lề trái.
    - **topmargin** : Canh lề trên

HTML cơ bản...

- Headings được định dạng với các thẻ: ***<h1>, <h2>, <h3>, <h4>, <h5>, <h6>***
  - Trước và sau mỗi tiêu đề văn bản tự động xuống dòng
  - Kích thước nhỏ dần từ h1 đến h6
  - Sử dụng làm tiêu đề cho bài viết
  - Thuộc tính:
    - **align = "căn chỉnh lề ":** giá trị: "left", "right", "center", "justify"

HTML cơ bản...

- **Các thẻ định dạng văn bản, ký tự…**
  - **...:** định dạng font chữ cho văn bản
    - Thuộc tính: **ace** = “tên font chữ” : .VnTime, Times New Roman, Arial
      - **Size** = ”kích thước” : giá trị 1->7 mặc định là 3
      - **Color** = “màu chữ”
        - Viết bằng tên tiếng Anh (red, blue,…)
        - Viết dạng #RRGGBB, RR, GG, BB ở dạng hexa. Ví dụ: #FFFFFF: Trắng, #FF0000: đỏ,…

HTML cơ bản...

- **Các thẻ định dạng văn bản, ký tự**

Tên thẻ	Định dạng
...	chữ đậm
...:	Chữ nghiêng
<u>....</u>	Chữ gạch chân
<big>..</big>	Chữ to
...	Chữ nhỏ
^{...}	Chỉ số trên
_{...}	Chỉ số dưới
...	Nhấn mạnh in đậm
...	Nhấn mạnh in nghiêng

HTML cơ bản...

- **Các thẻ định dạng văn bản, ký tự…**

Tên thẻ	Định dạng
<mark>..</mark>	Đánh dấu, highligh
~~...~~	Chữ bị gạch ngang
<address>...</address>	Thông tin tác gia, địa chỉ
<code>....</code>	Computer code
<kbd>....</kbd>	Keyboard input
<pre>...</pre>	Văn ban định dạng trước
<var>...</var>	Biến trong toán học
<q>...</q>	Quote

HTML cơ bản…

- **Phân đoạn và ngắt quãng văn bản**
  - Thẻ ** .. **: định dạng đoạn văn bản
    - Thuộc tính: *align* **= "căn chỉnh lề ":** giá trị: "*left*", "*right*", "*center*", "*justify*"
  - Thẻ
: sang dòng mới
  - Thẻ .. : Định dạng hiển thị giữa trang
  - Thẻ … , …: Tạo khối cho văn bản
    - Thuộc tính: *align* **= "căn chỉnh lề ":** giá trị: "*left*", "*right*", "*center*", "*justify*"
  - Thẻ <hr>: Tạo đường kẻ ngang
    - Thuộc tính:
      - *Align*: canh lề với giá trị center, right, left
      - *Width*: chỉ độ dài của đường thẳng: giá trị bằng pixel hoặc %. Mặc định 100%
      - *Size*: chỉ độ dày của đường thẳng
      - *Noshade*: chỉ đường thẳng được hiển thị bằng màu đặc thay vì có bóng

HTML cơ bản…

- **Danh sách:**
  - Danh sách dùng để liệt kê các phần tử
  - Một danh sách có nhiều phần tử, mỗi phần tử có thể là một danh sách con
  - Có 3 loại thể hiện danh sách
    - *Danh sách có thứ tự (**o**rdered **l**ist - ol)*
    - *Danh sách không có thứ tự (**u**nordered **l**ist -ul)*
    - *Danh sách mô tả (**d**escription **l**ist – dl)*

HTML cơ bản…

- Danh sách không có thứ tự (**U**nordered **L**ist) tạo các danh sách các mục có bullet
  - Thẻ **
….
**
  - Các phần tử: thẻ **
….
**
  - Thuộc tính: *Type*=“kiểu bullet”: square, circle, disc, none

```
Các thuộc tính của thẻ body
<ul type="circle">
    <li>background</li>
    <li>bgcolor</li>
    <li>text</li>
</ul>
```

Các thuộc tính của thẻ body

- background
- bgcolor
- text

HTML cơ bản…

- Danh sách có thứ tự (**O**rdered **L**ist) tạo các danh sách các mục có đánh số thứ tự
  - Thẻ **
….
**
  - Các phần tử: thẻ **
….
**
  - Thuộc tính:
    - *Type*=“kiểu đánh thứ tự”: 1, A, a, i, I
    - *start* = “giá trị bắt đầu”: áp dụng với type=“1”

```
<ol type="1" start="4">
    <li>Nguyễn Văn An</li>
    <li>Trần Văn Bình</li>
    <li>Đỗ Hùng Cường</li>
</ol>
```

4. Nguyễn Văn An
5. Trần Văn Bình
6. Đỗ Hùng Cường

HTML cơ bản...

- Danh sách mô tả (**D**escription **L**ist) tạo các danh sách với phần mô tả ở mỗi phần tử
  - Thẻ **<dl>....</dl>**
    - Các phần tử:
      - thẻ **<dt>....</dt>** tiêu đề
      - thẻ **<dd>...</dd>** mô tả

```
<dl>
  <dt>Coffee</dt>
  <dd>- black hot drink</dd>
  <dt>Milk</dt>
  <dd>- white cold drink</dd>
</dl>
```

Coffee
- black hot drink

Milk
- white cold drink

HTML cơ bản...

- **Thiết lập liên kết với thẻ <a>**
  - *Hyperlink* (liên kết siêu văn bản) là một đối tượng trên trang web hoặc tài liệu, nó có thể là văn bản, hình ảnh hoặc một đối tượng đa phương tiện
  - Khi nhắp chuột vào *hyperlink* trình duyệt web sẽ tự động chuyển đến vị trí đã được liên kết như: Trang web, Tải file, Gửi email, Chuyển đến một vị trí khác trên cùng trang web chứa hyperlink
  - Cú pháp:

```
<a href="url của liên kết">Mô tả liên kết</a>
```

HTML cơ bản...

- **Thiết lập liên kết với thẻ <a>...**
  - **Thuộc tính** *target*: *Sử dụng khi muốn tùy chọn cách mở liên kết*
    - _self : mở liên kết trên cùng cửa sổ/tab (mặc định)
    - _blank : mở liên kết trên cửa sổ/tab mới

```
<a href=http://www.diendancntt.com/ target="_blank"> Diễn đàn CNTT </a>
```

HTML cơ bản...

- **Thiết lập liên kết với thẻ <a>...**
  - **Đường dẫn tuyệt đối:** *Sử dụng trong trường hợp liên kết bên ngoài website*

```
<a href="http://www.diendancntt.com/"> Diễn đàn CNTT </a>
```

  - **Đường dẫn tương đối:** *Sử dụng trong trường hợp liên kết tới trang web cùng website*

```
<a href="diendancntt.html">Diễn đàn CNTT </a>
<a href="subfolder/page1.html">Diễn đàn CNTT </a>
<a href="../index.html">Diễn đàn CNTT </a>
```

HTML cơ bản…

- **Thiết lập liên kết với thẻ <a>…**
  - **Liên kết trong**: *Sử dụng trong trường hợp liên kết tới một vị trí cụ thể trên cùng trang web*

```
<a href=“#footer”>Xuống cuối trang</a>
```

  - **Trong đó***: footer là một vị trí cuối trang web được định nghĩa từ trước như sau:*

```
<h4 id="footer">Bạn đang ở cuối trang</h4>
```

HTML cơ bản...

- **Thiết lập liên kết với thẻ <a>...**
  - **Liên kết email**: *Sử dụng trong trường hợp muốn tạo liên kết để mở ứng dụng gửi email*

```
<a href="mailto:someone@example.com">Send email</a>
```

  - **Liên kết mobile:** *Sử dụng trong trường hợp liên kết tới ứng dụng gọi điện trên thiết bị và tự động gọi đến số được điền trong thẻ a*

```
<a href="tel:+84966554123">Hotline</a>
```

HTML cơ bản…

- **Thiết lập liên kết với thẻ <a>…**
  - Liên kết tới các loại file khác nhau
    - *Liên kết đến file usermanual.pdf*

```
<a href=“đường dẫn tới file/usermanual.pdf"> Đọc hướng dẫn sử dụng</a>
```

    - *Liên kết đến file logo.jpg*

```
<a href=“đường dẫn tới file/logo.jpg"> Xem ảnh</a>
```

HTML cơ bản…

- **Thẻ (tag) - Một số lưu ý**
  - Mọi khoảng trống, dấu xuống dòng trong HTML được thể hiện trên trang web là **1 khoảng trống duy nhất**
  - Để gõ một số ký tự đặc biệt ta phải sử dụng mã:
    - Khoảng trống (trong trường hợp muốn có nhiều hơn 1 ký tự trống):
    - Dấu nhỏ hơn (<) và lớn hơn (>): < và >
    - Dấu ngoặc kép (“): "
    - Ký hiệu ©: ©
    - …
  - Ghi chú trong HTML

```
<!-- Ghi nội dung chú thích ở đây-->
```

Phần tử ngữ nghĩa trong HTML

- ***Phần tử ngữ nghĩa là các phần tử mà tên của nó mang ý nghĩa rõ ràng***
  - *Ví dụ: <table>, <image>*
- **Một số thẻ ngữ nghĩa trong:**

Header, footer, article, section, details,

nav, aside, main, figure, summary, time

Phần tử ngữ nghĩa trong HTML...

- **header**: tiêu đề cho trang hoặc section, thường sử dụng header để chứa các nội dung cho phần đầu của trang web

```
<header>
 <div class="logo">
    <img src="logo.png" alt="Logo">
</div>
<div class="bg_header"></div>
<h1>ABC Studio</h1>
<nav>
  <ul>
      <li><a href="#">Trang chủ</a></li>
      <li><a href="#">Giới thiệu</a></li>
      <li><a href="#">Sản phẩm</a></li>
      <li><a href="#">Liên hệ</a></li>
  </ul>
 </nav>
<div class="login">
     <a href="#">Đăng nhập</a>
     <a href="#">Đăng ký</a>
</div>
</header>
```

Phần tử ngữ nghĩa trong HTML…

- **nav**: Sử dụng để chứa một danh sách các liên kết điều hướng
  - **Lưu ý**: thẻ nav chỉ sử dụng với khối liên kết điều hướng chính của trang web, một số liên kết khác không nhất thiết phải đưa vào nav

```
<nav>
<a href="#">Trang chủ</a>
<a href="#">Giới thiệu</a>
<a href="#">Sản phẩm</a>
<a href="#">Liên hệ</a> </nav>
```

Phần tử ngữ nghĩa trong HTML…

- **section**: nếu trang web cho chia thành nhiều phần nội dung riêng biệt thì <section> sử dụng để xác định một phần nội dung cho trang web

```
<section>
<h2>Sản phẩm nổi bật</h2>
<p>Đây là nội dung của phần này </p>
<p>Nội dung này có thể bao gồm các đoạn văn, hình ảnh, hoặc các phần tử khác của trang web.</p>
</section> <section>
<h2>Sản phẩm được quan tâm nhiều</h2> <p>Đây là nội dung của phần
</p>
</section>
```

Phần tử ngữ nghĩa trong HTML…

- **footer**: sử dụng để xác định phần chân (footer) cho trang web hoặc một section. *Footer thường bao gồm các thông tin sau*:
  - Thông tin tác giả
  - Thông tin bản quyền
  - Địa chỉ liên hệ
  - Sơ đồ trang web
  - Liên kết tới đầu trang
  - …

Phần tử ngữ nghĩa trong HTML…

- **main**: sử dụng để chứa nội dung chính (phần thân) của trang web
- **article**: sử dụng để chứa nội dung độc lập. Ví dụ:
  - Bài viết trên forum, blog
  - Bình luận của người dùng
  - Bài báo
  - …

Phần tử ngữ nghĩa trong HTML…

- **details và summary**: sử dụng để tạo tiêu đề, hiển thị và thu gọn nội dung chi tiết

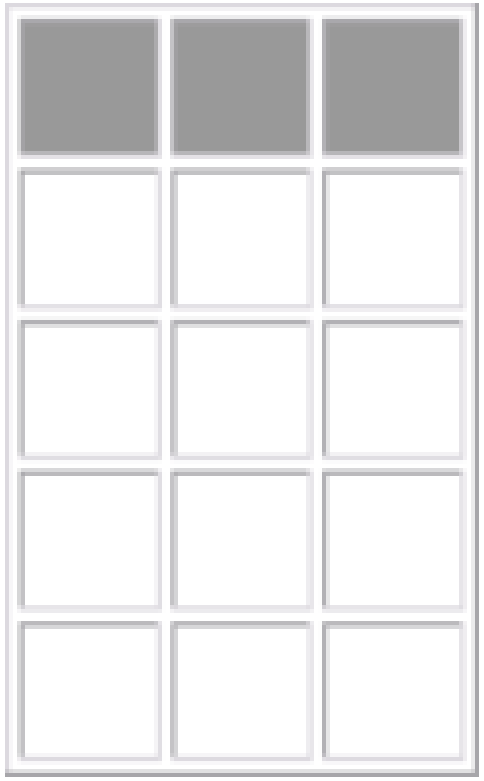
```
<details>
<summary>Click to show details</summary> <ul>
<li>Chương 1. Giới thiệu</li>
<li>Chương 2. Cơ bản về HTML5</li>
<li>Chương 3. CSS</li>
</ul> </details>
```

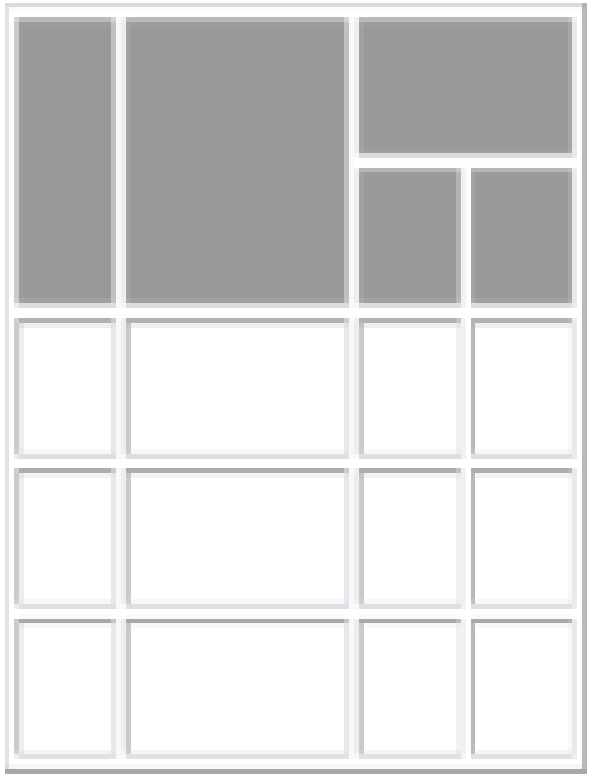
Phần tử ngữ nghĩa trong HTML…

- **aside**: sử dụng để chứa các thông tin ngoài nội dung chính, như: menu sườn, quảng cáo, tin tức…

Bảng biểu (table)

- HTML coi một *bảng gồm nhiều dòng*, một *dòng gồm nhiều ô*, và *chỉ có ô mới chứa dữ liệu* của bảng





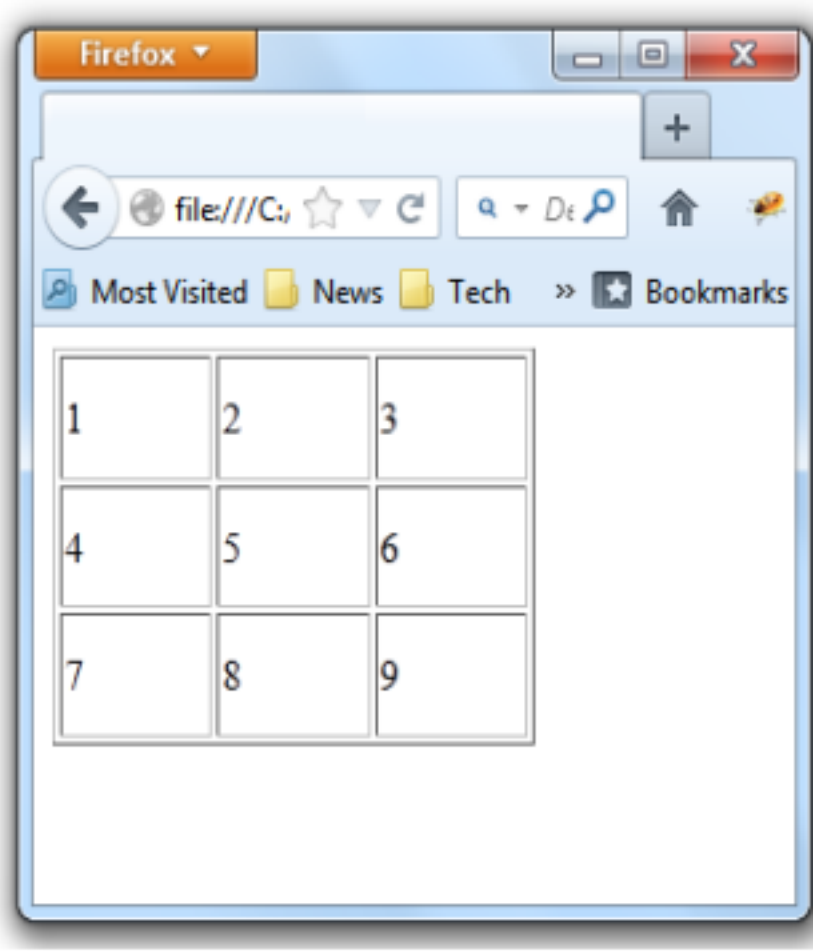
Bảng biểu (table)…

- **Các thẻ**
  - Tạo bảng <table>…</table>: Mỗi bảng chỉ có 1 cặp thẻ này.
  - Tạo dòng <tr>…</tr>: Bảng có bao nhiêu dòng thì có bấy
  - nhiêu cặp thẻ này
  - Tạo ô:
    - Ô tiêu đề của bảng: <th>…</th>
    - Ô dữ liệu: <td>…</td>
      - Tổng số thẻ <td> và <th> bằng số ô của bảng. Dòng có bao nhiêu ô thì có bấy nhiêu thẻ <td> và/hoặc <th> nằm trong cặp thẻ <tr>…</tr> tương ứng
    - Lưu ý: Để có được một ô trống trong bảng (ô không có dữ liệu) thì cần đặt nội dung ô là:

Bảng biểu (table)…

- Ví dụ:

```
<table border="1" width="200" height="150">
    <tr>
        <td>1</td>
        <td>2</td>
        <td>3</td>
    </tr>
    <tr>
        <td>4</td>
        <td>5</td>
        <td>6</td>
    </tr>
    <tr>
        <td>7</td>
        <td>8</td>
        <td>9</td>
    </tr>
</table>
```



Bảng biểu (table)…

- **Thuộc tính của thẻ <table>:**
  - *border*=“số”: kích thước đường viền. Đặt bằng 0 (mặc định): không có đường viền.
  - *width*=“rộng”, *height*=“cao”: độ rộng và độ cao của bảng. Có thể đặt theo 2 cách:
    - n: (n là số) Quy định độ rộng, cao là n pixels
    - n%: Quy định độ rộng, cao là n% độ rộng, cao của đối tượng chứa bảng.
  - *cellspacing*=“số”: Khoảng cách giữa 2 ô liên tiếp
  - *cellpadding*=“số”: Khoảng cách từ border ô đến nội dung ô
  - *bgcolor*=“màu”: màu nền của bảng
  - *background*=“địa_chỉ_ảnh”: Địa chỉ của file ảnh làm nền cho bảng. Nên sử dụng đường dẫn tương đối nếu có thể

Bảng biểu (table)…

- **Thuộc tính của thẻ <td>, <th>:**
  - *bgcolor*=“màu”: màu nền của ô
  - *background*=“địa_chỉ_ảnh”: Địa chỉ của file ảnh làm nền cho ô. Nên sử dụng đường dẫn tương đối nếu có thể.
  - *width*=“rộng”, *height*=“cao”: độ rộng và độ cao của ô. Có thể đặt theo 2 cách:
    - n: (n là số) Quy định độ rộng, cao là n pixels
    - n%: Quy định độ rộng, cao là n% độ rộng, cao của bảng.
  - *align*=“căn_lề”: cách căn chỉnh dữ liệu trong ô theo chiều ngang: left, right, center, justify.
  - *valign*=“căn lề đứng”: cách căn chỉnh dữ liệu trong ô theo chiều đứng: top, middle, bottom.
  - *colspan*=“số”: số cột mà ô này chiếm (mặc định là 1)
  - *rowspan*=“số”: số dòng mà ô này chiếm (mặc định là 1)
  - *nowrap*: nếu có sẽ làm cho dữ liệu trong ô không tự xuống dòng

Bảng biểu (table)…

- Thẻ <caption> : chứa phụ đề của bảng

```
<table>
  <caption>Monthly savings</caption>
  <tr>
    <th>Month</th>
    <th>Savings</th>
  </tr>
  <tr>
    <td>January</td>
    <td>$100</td>
  </tr>
</table>
```

Monthly savings

Month	Savings
January	$100
February	$50

Bảng biểu (table)…

- Nhóm Cột:
  - Thẻ <colgroup> nhóm các cột thành từng nhóm giúp định dạng các cột cùng lúc
    - Thẻ colgroup đứng sau <caption>, trước <thead>,<tbody>, <tfoot>, <tr>

```
<table>
  <colgroup>
    <col span="2" style="background-color:red">
    <col style="background-color:yellow">
  </colgroup>
  <tr>
    <th>ISBN</th>
    <th>Title</th>
    <th>Price</th>
  </tr>
  <tr>
    <td>3476896</td>
    <td>My first HTML</td>
    <td>$53</td>
  </tr>
</table>
```

ISBN	Title	Price
3476896	My first HTML	$53
5869207	My first CSS	$49

Bảng biểu (table)…

- Thẻ <thead>: định nghĩa phần đầu của bảng
- Thẻ <tbody>: định nghĩa phần thân của bảng
- Thẻ <tfoot>: định nghĩa phần cuối của bảng

Month	Savings
January	$100
February	$80
Sum	$180

```
<table>
 <thead>
  <tr>
    <th>Month</th>
    <th>Savings</th>
  </tr>
 </thead>
 <tfoot>
  <tr>
    <td>Sum</td>
    <td>$180</td>
  </tr>
 </tfoot>
 <tbody>
  <tr>
    <td>January</td>
    <td>$100</td>
  </tr>
  <tr>
    <td>February</td>
    <td>$80</td>
  </tr>
 </tbody>
</table>
```

Form

- **Các đối tượng nhập dữ liệu:**
  - Cho phép người sử dụng nhập dữ liệu trên trang web. Dữ liệu này có thể được gửi về server để xử lý
  - Người sử dụng nhập dữ liệu thông qua các điều khiển (controls). Có nhiều loại control:
    - Form
    - Oneline Textbox
    - Checkbox
    - Radio Button
    - Button
    - Combo box (drop-down menu)
    - Listbox
    - Hộp nhập văn bản nhiều dòng (TextArea)
    - …

Form..

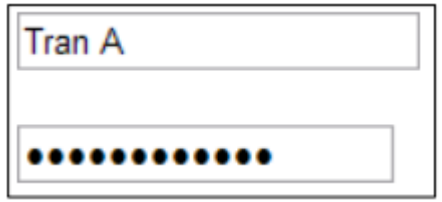
- **Các đối tượng nhập dữ liệu…**
  - Tất cả các điều khiển đều có tên được quy định qua thuộc tính **name**. Tuy nhiên có một số điều khiển thì name không quan trọng (các điều khiển mà sau này không cần lấy dữ liệu)
  - Các điều khiển từ số 2. đến số 5 được định nghĩa nhờ thẻ **<input>** và thuộc tính **type** sẽ xác định là điều khiển nào sẽ được tạo ra.
- **Form:**
  - Sử dụng để **chứa mọi đối tượng khác**
  - Không nhìn thấy khi trang web được hiển thị
  - Quy định một số thuộc tính quan trọng như **method**, **action**

Form..

- **Form…**
  - Thẻ tạo form: **<form>…</form>**
  - Các thuộc tính:
    - *name*=“tên_form”: Không quan trọng lắm
    - *action*=“địa chỉ nhận dữ liệu”: Nên sử dụng đường dẫn tương đối nếu nằm trong cùng 1 web
    - method=“phương thức gửi dữ liệu”. Chỉ có 2 giá trị: GET (mặc định), POST

Form..

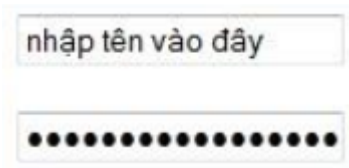
- **Hộp nhập văn bản 1 dòng (Oneline Textbox):**
  - Sử dụng để nhập các văn bản ngắn (trên 1 dòng) hoặc mật khẩu
  - Thẻ: **<input>**
  - Thuộc tính:
    - *name*=“tên_đt”: quan trọng
    - *type*=“text”:Ô nhập văn bản thường
    - *type*=“password”: ô nhập mật khẩu
    - *value*=“giá trị mặc định”



Form..

- **Hộp nhập văn bản 1 dòng (Oneline Textbox):..**
  - Có nhiều *type* khác nhau:
    - Color
    - Date
    - Datetime
    - File
    - Email
    - hidden

```
<form action="user.php" method="get">
    <input type="text" name="ten" value="nhập tên vào đây">
    <br><br>
    <input type="password" name="matkhau" value="nhập mật khẩu vào đây">
</form>
```

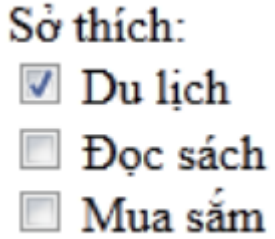


Form…

- **Checkbox:**
  - Cho phép **chọn nhiều** lựa chọn trong một nhóm lựa chọn được đưa ra bằng cách đánh dấu (“tích”).
  - Thẻ: **<input>**: mỗi ô nhập cần 1 thẻ
  - Thuộc tính
    - name=“tên_đt”: quan trọng
    - type=“checkbox”
    - value=“giá trị”: đây là giá trị chương trình sẽ nhận được nếu NSD chọn ô này.
    - *checked*: nếu có thì nút này mặc định được chọn

```
<form action="dangky.html" method="get">
Sở thích: <br>
    <input name="sothich[]" checked="checked" type="checkbox" value="du lich"> Du lịch
    <br>
    <input name="sothich[]" type="checkbox" value="doc sach"> Đọc sách
    <br>
    <input name="sothich[]" type="checkbox" value="mua sam"> Mua sắm

<form>
```



Form…

- **Radio Button:**
  - Cho phép **chọn một** lựa chọn trong một nhóm lựa chọn được đưa ra
  - Trên 1 form có thể có nhiều nhóm lựa chọn kiểu này
  - Thẻ: **<input>**: Mỗi ô cần 1 thẻ
  - Thuộc tính:
    - *name*=“tên_đt”: quan trọng. Các đối tượng cùng tên thì thuộc cùng nhóm.
    - *type*=“radio”
    - *value*=“giá trị”: đây là giá trị chương trình sẽ nhận được nếu NSD chọn ô này.
    - *checked*: nếu có thì nút này mặc định được chọn

```
<form action="dangky.html" method="get">
Giới tính: <br>
    <input name="gioitinh" checked type="radio" value="nam"> Nam
    <br>
    <input name="gioitinh" type="radio" value="nu"> Nu

<form>
```

Giới tính:
Nam
Nu

Form…

- **Button**:
  - Sử dụng để người dùng ra lệnh thực hiện công việc
  - Trên web có 3 loại nút:
    - *submit*: Tự động ra lệnh gửi dữ liệu
    - *reset*: đưa mọi dữ liệu về trạng thái mặc định
    - *normal*: người lập trình tự xử lý
  - Thẻ: **<input>**
  - Thuộc tính
    - *name*="tên_ĐT": thường không quan trọng
    - *type*="submit": nút submit
    - *type*="reset": nút reset
    - *type*="button": nút thông thường (normal), it sử dụng.
    - *value*="tiêu đề nút"

Form…

- **Combobox/Drop-down Menu**:
  - Bao gồm một danh sách có nhiều phần tử. Tại một thời điểm
  - chỉ có 1 phần tử được chọn
  - NSD có thể chọn 1 phần tử trong danh sách xổ xuống bằng
  - cách kích vào mũi tên bên phải hộp danh sách.
  - Thẻ tạo hộp danh sách:

  <select>Danh sách phần tử</select>

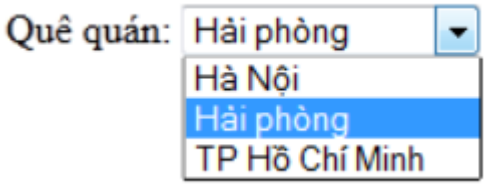
  - Thuộc tính:
    - *name*=“tên_ĐT”: quan trọng
  - Thẻ tạo 1 phần tử trong danh sách:

  <option>Tiêu đề phần tử</option>

Form…

- **Combobox/Drop-down Menu**:
  - Thuộc tính:
    - *value*=“giá trị”: giá trị chương trình nhận được nếu phần tử được chọn
    - selected: nếu có thì phần tử này mặc định được chọn

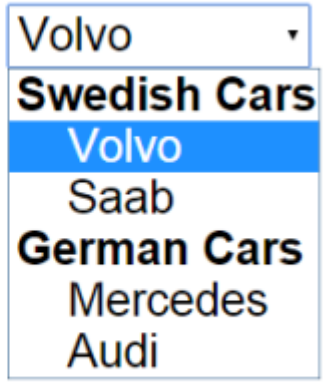
```
<form action="dangky.html" method="get">
Quê quán:
<select name="quequan">
    <option value="hanoi">Hà Nội</option>
    <option selected value="haiphong">Hải phòng</option>
    <option value="hochiminh">TP Hồ Chí Minh</option>
</select>
<form>
```



Form…

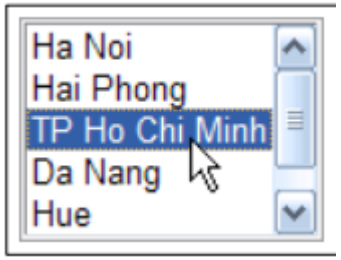
- **Combobox/Drop-down Menu**:
  - Thẻ **<optgroup>** được dùng để nhóm các lựa chọn thành nhóm

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



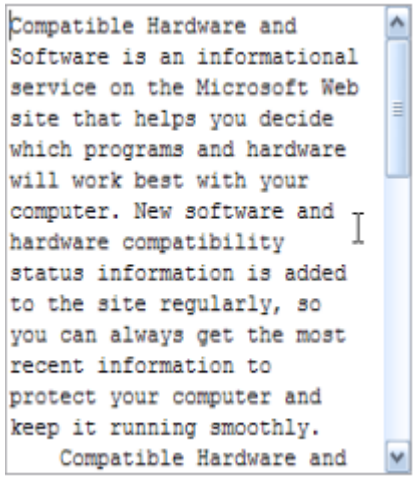
Form…

- **Listbox:**
  - Tương tự như Combo box, tuy nhiên có
  - thể nhìn thấy nhiều phần tử cùng lúc, có
  - thể lựa chọn nhiều phần tử
  - Thẻ: <select>…</select>
  - Thuộc tính: tương tự của combo tuy nhiên
  - có 2 thuộc tính khác:
    - *size*=“số dòng”
    - *multiple*: cho phép lựa chọn nhiều phần tử cùng lúc
  - Thẻ <option>…</option> tương tự của combobox



Form…

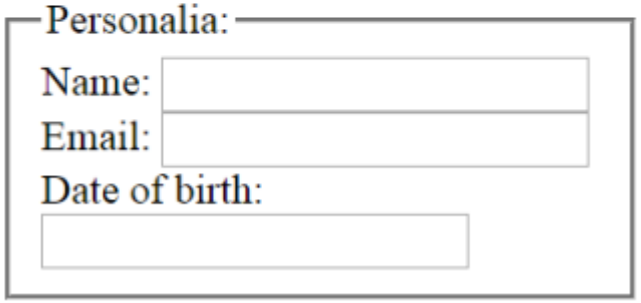
- Hộp nhập văn bản nhiều dòng
  - Cho phép nhập văn bản dài trên nhiều dòng.
  - Thẻ: <textarea>
    //Nội dung mặc định
    </textarea>
  - Thuộc tính:
    - *name*=“tên_ĐT”: quan trọng
    - *rows*=“số dòng”
    - *cols*=“số cột”
    - Lưu ý: rows tính theo số dòng văn bản, cols tính theo số ký tự chuẩn trên dòng.



Form…

- Thẻ **<fieldset>** nhóm các đối tượng trong form thành nhóm
- Thẻ **<legend>** phụ đề chung cho nhóm đối tượng

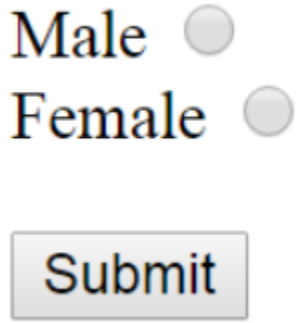
```
<form>
  <fieldset>
    <legend>Personalia:</legend>
    Name: <input type="text"><br>
    Email: <input type="text"><br>
    Date of birth: <input type="text">
  </fieldset>
</form>
```



Form…

- Thẻ **<label>** chứa tiêu đề cho các đối tượng input
  - Khi nhấn vào tiêu đề sẽ tự động chọn input tương ứng

```
<form action="demo_form.asp">
  <label for="male">Male</label>
  <input type="radio" name="sex" id="male" value="male"><br>
  <label for="female">Female</label>
  <input type="radio" name="sex" id="female" value="female"><br>
  <input type="submit" value="Submit">
</form>
```



Form…

- **Khung (Frame):**
  - Cho phép chia một trang web làm nhiều phần, mỗi phần chứa nội dung của 1 trang web khác
  - Trình duyệt có thể không hỗ trợ khung
  - Tạo trang web chứa các khung: Thay thẻ <body>…</body> bằng

```
<frameset>
các khung
</frameset> <noframes>
nội dung trong trường hợp trình duyệt
không hỗ trợ khung
</noframes>
```

Form…

- **Frame:…**
  - Một số thuộc tính của <frameset>
    - *rows* = “n1, n2, … nk” hoặc **cols** = “n1, n2, … nk”: Quy định có k dòng (hoặc cột), độ rộng dòng (cột) thứ i là ni. ni là số, có thể thay bằng *: phần còn lại
    - *frameborder* = yes hoặc no
    - *framespacing* = “n”: Khoảng cách giữa 2 khung
  - Tạo 1 khung có nội dung là 1 trang web nào đó: **<frame>:**
    - Thuộc tính:
      - *src*=“Địa chỉ chứa nội dung”
      - *name*=“tên khung”
      - *noresize*: Không được thay đổi kích thước
  - Thẻ **<base>** mặc định
    - Thuộc tính
      - *target*=“Cửa sổ mặc định”
      - *href*=“Địa chỉ gốc mặc định”

Form…

- **Iframes**: Sử dụng để hiển thị 1 trang web trong một trang web khác

```
<iframe src="URL"></iframe>
```

  - **Thuộc tính**
    - *url*: địa chỉ trang web con muốn hiển thị
    - *Width*: độ rộng
    - *Height*: độ cao
    - *Frameborder*: đường viền

```
<iframe width="420" height="315"
src="https://www.youtube.com/embed/tgbNymZ7vqY?playlist=tgbNymZ7vqY&loop=
1"> </iframe>
```

Form…

- **Đa phương tiện**
  - Âm thanh nền: **<bgsound>** :
    - Thuộc tính:
      - *src*=“địa chỉ file âm thanh”
      - *loop*=“n”: số lần lặp. -1: mặc định: mãi mãi.
  - Flash: Thẻ **<object>** chèn file Flash

```
<object width="400" height="400" data="helloworld.swf"></object>
```

    - **Thuộc tính**
      - *Width*: độ rộng
      - *Height*: độ cao
      - *Data*: địa chỉ file Flash

Form…

- **Đa phương tiện…**
  - Thẻ <video> trong HTML5 cung cấp một cách tiêu chuẩn để nhúng video vào trang web

```
<video width="320" height="240" controls>
<source src="movie.mp4" type="video/mp4">
<source src="movie.ogg" type="video/ogg"> Your browser does not support the video tag.
</video>
```

  - Thẻ <audio> trong HTML5 cung cấp một cách tiêu chuẩn để nhúng âm thanh vào trang web.

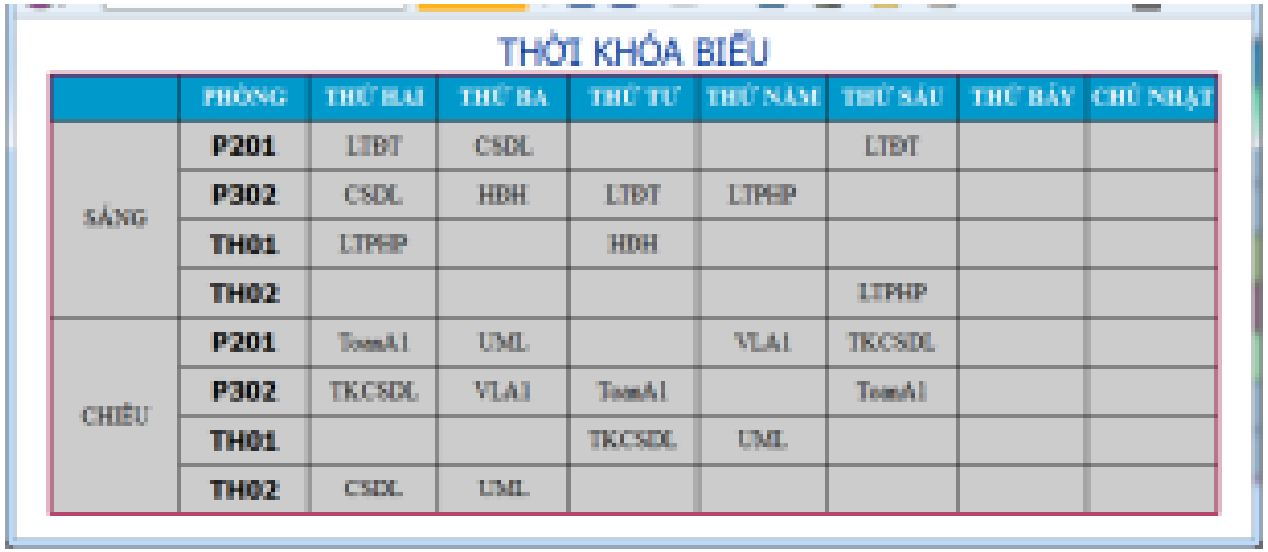
```
<audio controls>
<source src="horse.ogg" type="audio/ogg">
<source src="horse.mp3" type="audio/mpeg">
Your browser does not support the audio element.
</audio>
```

2. CSS

Giới thiệu CSS

- CSS (**C**ascading **S**tyle **S**heets): Là một ngôn ngữ dùng để trình bày hình thức thể hiện của các phần tử HTML: định dạng các phần tử văn bản (cơ chữ, font chữ, màu sắc ...), bố cục, dàn trang ...
  - *Cascading*: thể hiện cách ảnh hưởng của nhiều mã CSS đến phần tử HTML. Các CSS tác dụng vào một phần tử HTML xếp chồng thành nhiều lớp. Kết quả là sự tổng hợp của các lớp đó.
  - *Style Sheets*: mã điều khiển cách hiện thị nội dung trang web.
  - Tái sử dụng cho các trang web khác
- CSS và HTML luôn đi cùng nhau: **HTML** tạo ra cấu trúc, nội dung các phần tử trong trang, **CSS** quy định cách hiện thị phần tử trên trang.

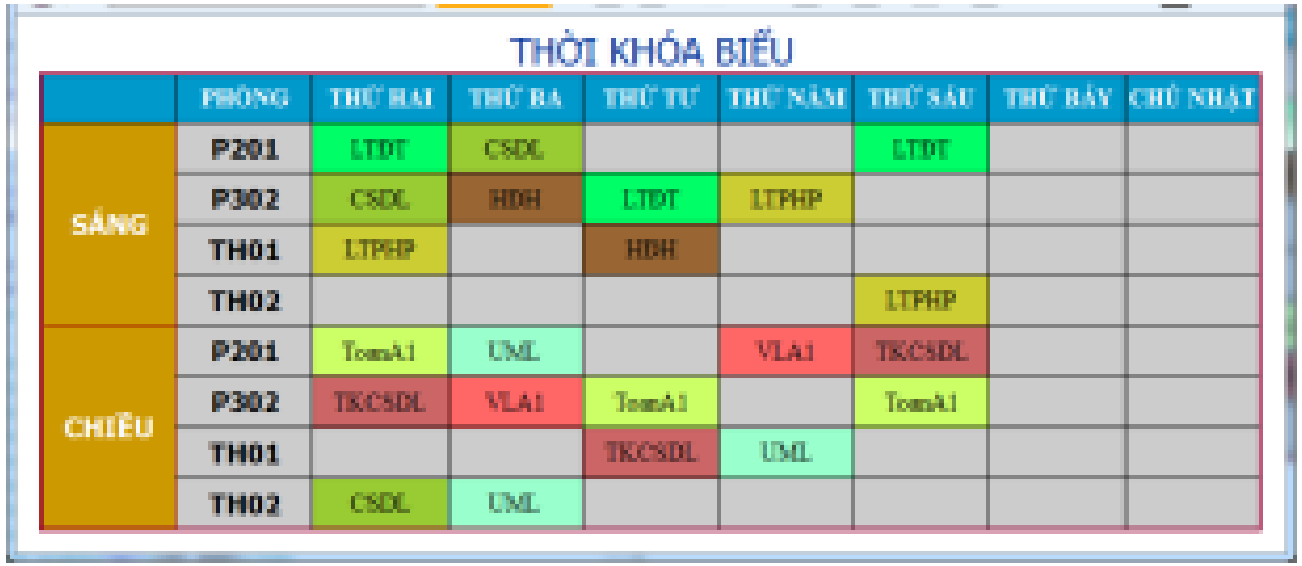
Giới thiệu CSS…



THỜI KHÓA BIỂU

	PHÒNG	THỨ HAI	THỨ BA	THỨ TƯ	THỨ NĂM	THỨ SÁU	THỨ BẢY	CHỦ NHẬT
SÁNG	P201	LTĐT	CSDL			LTĐT		
	P302	CSDL	HĐH	LTĐT	LTPHP			
	TH01	LTPHP		HĐH				
	TH02					LTPHP		
CHIỀU	P201	ToanA1	UML		VLA1	TKCSDL		
	P302	TKCSDL	VLA1	ToanA1		ToanA1		
	TH01			TKCSDL	UML			
	TH02	CSDL	UML					

HTML



THỜI KHÓA BIỂU

	PHÒNG	THỨ HAI	THỨ BA	THỨ TƯ	THỨ NĂM	THỨ SÁU	THỨ BẢY	CHỦ NHẬT
SÁNG	P201	LTĐT	CSDL			LTĐT		
	P302	CSDL	HĐH	LTĐT	LTPHP			
	TH01	LTPHP		HĐH				
	TH02					LTPHP		
CHIỀU	P201	ToanA1	UML		VLA1	TKCSDL		
	P302	TKCSDL	VLA1	ToanA1		ToanA1		
	TH01			TKCSDL	UML			
	TH02	CSDL	UML					

CSS

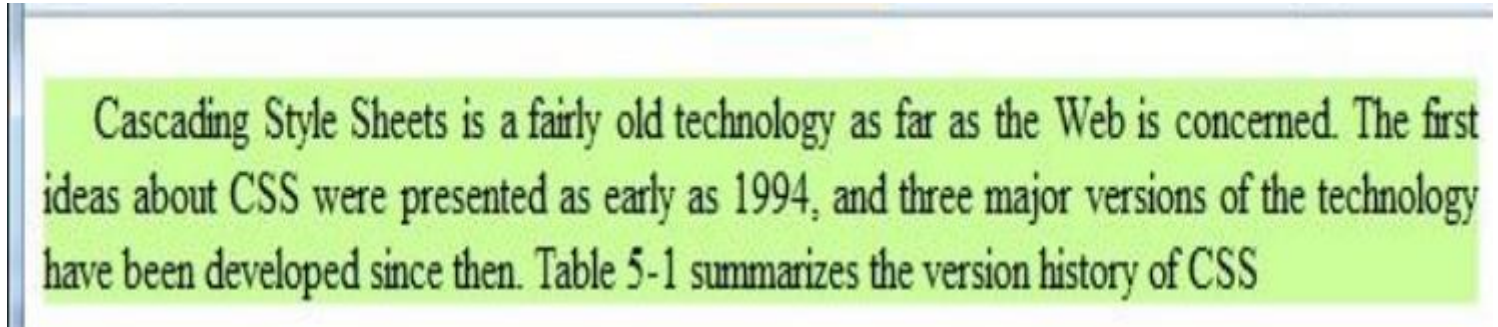
Định nghĩa CSS…

- Cú pháp định nghĩa CSS cho một Selector

```
SelectorName
{
        property 1: value1;
        property 2: value2;
        …
        property N: valueN;
}
```

  - Trong đó:
    - *SelectorName*: là tên các tag đã được định nghĩa trước trong HTML (tag <p>, tag <table>,..) hoặc tên do người dùng định nghĩa mới.
    - *Property*: tên các thuộc tính do CSS hỗ trợ
    - *Value*: giá trị ứng với các thuộc tính
  - *Lưu ý*: ghi chú trong CSS dùng */* … */*

```
...
<style type="text/css">
p {
        background-color:#CF9;
        text-indent:20px;
        text-align:justify;
}
</style>
<body>
        <p> Cascading Style Sheets is a fairly old
        technology as far as the Web is concerned. The
        first ideas about CSS were presented as early as
        1994, and three major versions of the
        technology have been developed since then.
        Table 5-1 summarizes the version history of
        CSS </p>
</body>
...
```



Phương pháp sử dụng CSS

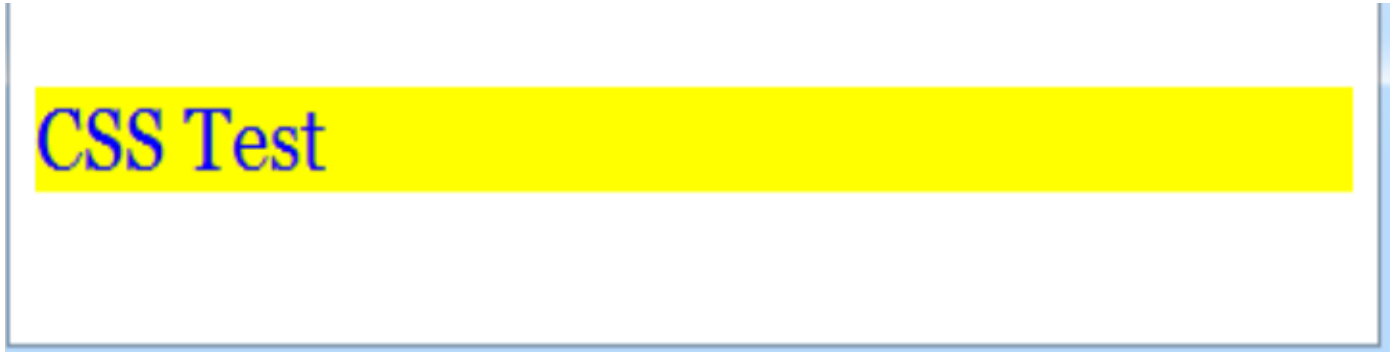
- **Inline Styles:** Các thuộc tính style được nhúng trực tiếp trong các thẻ (tag) khi sử dụng

```
…
<body>
<h1 style="font-size: 48px; font-family:Arial,
Helvetica, sans-serif;color: green;"> CSS Test
</h1>
</body>
</html>
```

Phương pháp sử dụng CSS…

- **Embedded Styles:** Các thuộc tính style cho các thẻ (tag) được khai báo trước trong phần tag <Head> của trang trước khi sử dụng

```
…
<style type="text/css">
<!--
p {font-size: 1.5em; font-family: Tahoma; color: blue;
background-color: yellow;} em {font-size: 2em; color:
green;}
-->
</style>
</head> <body>
<p>CSS Test</p>
</body>
</html>
```

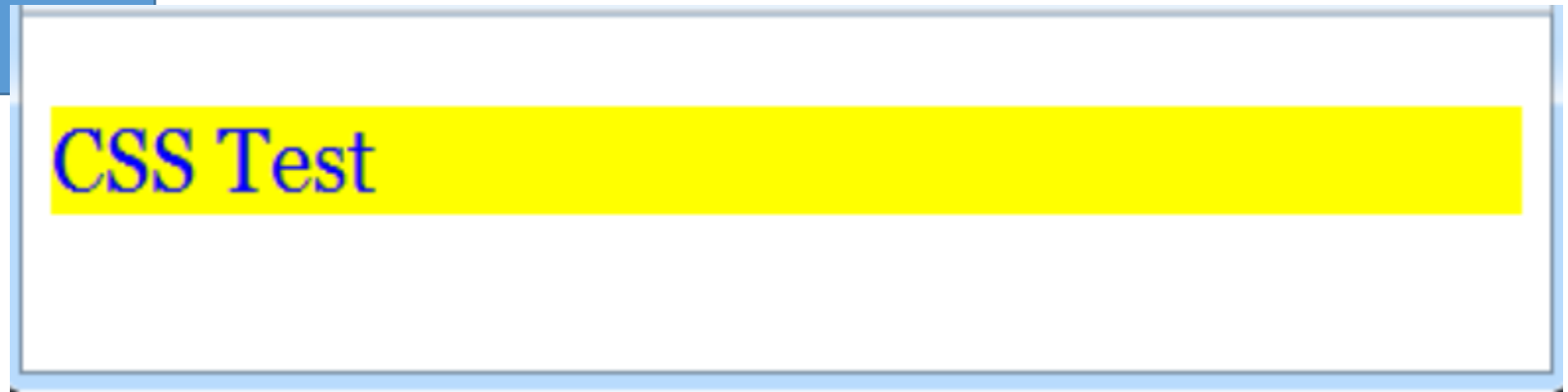


Phương pháp sử dụng CSS…

- **Imported Styles:** Các thuộc tính style cho các thẻ (tag) được nhúng từ một tập tin *.css bên ngoài vào trang.

```
…
<style type="text/css">
@import url("css/cssTestCSS.css");
</style>
</head>
<body>
<p>CSS Test</p>
</body>
</html>
```

```
@charset "utf-8"; /* CSS Document File:
cssTestCSS.css */
p {font-size: 1.5em; font-family: Tahoma;
color: blue; background-color: yellow;}
em {font-size: 2em; color: green;}
```

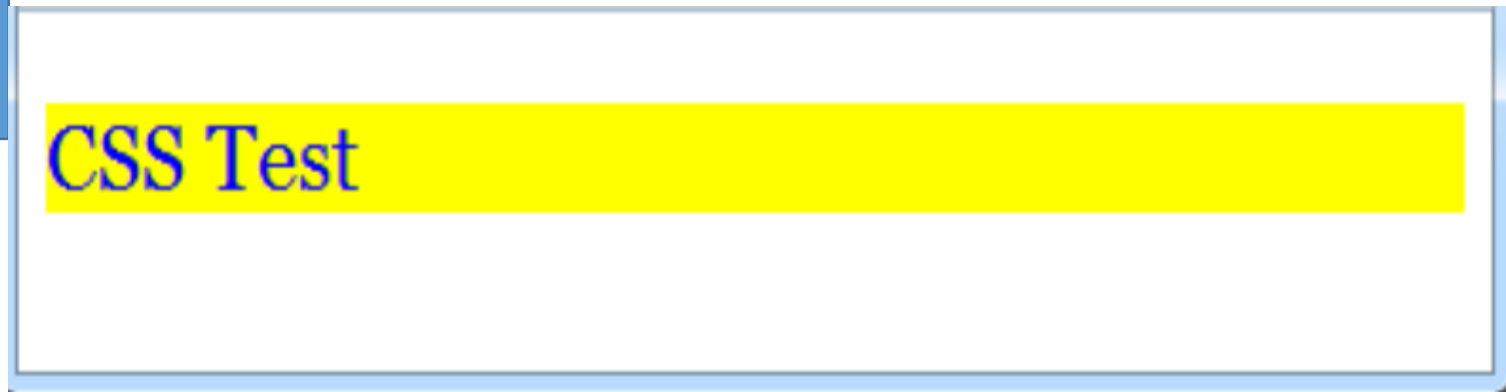


Phương pháp sử dụng CSS…

- **Linked Styles:** Các thuộc tính style cho các thẻ (tag) được nhúng từ một tập tin *.css bên ngoài vào trang (tương tự như phương pháp Imported Styles)

```
…
<link href="css/cssTestCSS.css" rel="stylesheet"
type="text/css" />
</head>
<body>
<p>CSS Test</p>
</body>
</html>
```

```
@charset "utf-8"; /* CSS Document File:
cssTestCSS.css */
p {font-size: 1.5em; font-family: Tahoma;
color: blue; background-color: yellow;}
em {font-size: 2em; color: green;}
```



Phương pháp sử dụng CSS…

- **Độ ưu tiên của các phương pháp**

```
…
<style type="text/css">
@import url("CSSImported.css");
</style>

<style type="text/css">
p {color:green;}
</style>
<link href="CSSLinked.css" rel="stylesheet" type="text/css" />
</head>
<body>
<p style="color:black"> CSS Test 1 </p>
<p> CSS Test 2 </p>
</body> </html>
```

```
/* File: CSSImported.css
*/ p {color: blue;}
```

```
/* File: CSSLinked.css */ p
{color: red;}
```

CSS Test 1

CSS Test 2

Phương pháp sử dụng CSS…

- Ưu điểm và nhược điểm của các phương pháp

ĐÁNH GIÁ	INLINE STYLES	EMBEDDED STYLES	IMPORTED STYLES	LINKED STYLES
Cú pháp	 Lập trình Web với PHP 	<style type=“text/css”> p {color=red;} </style> Lập trình Web với PHP	<style type="text/css"> @import url(“my.css"); </style> Lập trình Web với PHP	<link href=“my.css" rel="stylesheet" type="text/css" /> Lập trình Web với PHP
Ưu điểm	-Định nghĩa nhanh -Dễ quản lý cho từng tag trong một trang	-Định nghĩa nhanh -Dễ quản lý cho từng trang	-Có thể áp dụng style đồng bộ cho một site. -Thông tin Style được trình duyệt cache → cải thiện tốc độ duyệt web ở những lần sau.	
Nhược điểm	- Khó áp dụng đồng bộ cho từng tag trong cùng một trang	- Khó áp dụng đồng bộ cho các trang	- Tối ưu tập tin *.css để cải thiện tốc độ duyệt cho lần đầu tiên truy cập site.	

Phân loại các CSS selector

- **CSS Selector** là các phương pháp dùng để định dạng các thuộc tính cho một hay nhiều thẻ (tag) HTML đã có hoặc xây dựng các lớp (class) định dạng dùng áp dụng chung cho các thẻ (tag)
- Tùy theo phiên bản, CSS hỗ trợ tập các CSS Selector khá đa dạng

Phân loại các CSS selector…

- Bảng phân loại các CSS Selector thông dụng

LOẠI	MÔ TẢ	VÍ DỤ
element	Định dạng được áp dụng cho tất cả các tag element trong tài liệu Web.	h1{color:red} /*Tất cả các tag <h1> sẽ bị định dạng màu chữ là red*/
#ID	Định dạng được áp dụng cho tất cả các tag có thuộc tính ID trong tài liệu Web	#test {color: green;} */* Tất cả các tag có thuộc tính id="test" đều bị định dạng màu chữ là green */*
element#ID	Định dạng được áp dụng cho tất cả các tag element có thuộc tính ID trong tài liệu Web	h3#contact {color: red;} */* Tất cả các tag <h3> có thuộc tính id="contact" đều bị định dạng màu chữ là red*/*

Phân loại các CSS selector…

- Bảng phân loại các CSS Selector thông dụng

LOẠI	MÔ TẢ	VÍ DỤ
.class	Định dạng được áp dụng cho tất cả các tag có thuộc tính class trong tài liệu Web	.note {color: red;} /* Tất cả các tag có thuộc tính class="note" đều bị định dạng màu chữ là red*/
element.class	Định dạng được áp dụng cho tất cả các tag Element có thuộc tính class tương ứng	h1.note {text-decoration: underline;} /* Tất cả các tag <H1> có thuộc tính class=note sẽ bị định dạng gạch chân */
a:link a:active a:visited	Định dạng được áp dụng cho các liên kết trong tài liệu Web	a:link {font-weight: bold;} a:active {color: red;}a:visited {text-decoration: linethrough;}

Phân loại các CSS selector…

```
…
<title>Khoa CNTT</title>
<style type="text/css">
        p#HTTT{color:#06C;}
        p#MMT{color:#C00;}
</style>
</head>
<body>
<p id="HTTT"> HỆ THỐNG THÔNG TIN</p>
- TS Nguyễn Đình Thuân<br>
- ThS Nguyễn Thị Kim Phụng<br>
<p id="MMT"> MẠNG MÁY TÍNH</p>
- TS Đàm Quang Hồng Hải<br>
- TS Nguyễn Anh Tuấn<br>
</body>
</html>
```

Phân loại các CSS selector…

```
…
<title>Khoa CNTT</title>
<style type="text/css">
        .HTTT{color:#06C;}
        .MMT{color:#C00;}
</style>
</head>
<body>
<p class="HTTT"> HỆ THỐNG THÔNG TIN</p>
- TS Nguyễn Đình Thuân<br>
- ThS Nguyễn Thị Kim Phụng<br>
<p class="MMT"> MẠNG MÁYTÍNH</p>
- TS Đàm Quang Hồng Hải<br>
- TS Nguyễn Anh Tuấn<br>
</body>
</html>
```

Định dạng văn bản sử dụng CSS

- Tạo màu cho văn bản:
  - Cách biểu diễn thuộc tính màu
    - Cách 1: dùng tên màu, ví dụ: *red, orange…*
    - Cách 2: dùng giá trị HEX của màu, ví dụ: *#FF0000*
    - Cách 3: dùng giá trị RGB của màu, ví dụ: *rgb(255,0,0)*
  - Thuộc tính:
    - *color*: Màu chữ
    - *background-color*: Màu nền

```
body {
        background-color: lightblue;
        color: black;
}
h1 {
        color: #e7bc1e;
        background-color: black;
}
```

Định dạng văn bản sử dụng CSS…

- **Thuộc tính *padding***: Sử dụng để định khoảng cách giữa một phần tử và đường biên xung quanh nó
  - Ví dụ:
    - **Dạng shorthand:**
      - padding: 20px 10px 10px 20px;
    - **Dạng longhand:**
      - padding-top: 20px; padding-right: 10px; padding-bottom: 10px; padding-left: 20px;
- **Thộc tính *margin***: Sử dụng để định khoảng cách từ đường biên của một phần tử tới các phần tử khác
  - Ví dụ:
    - **Dạng shorthand:**
      - margin: 20px 10px 10px 20px;
    - **Dạng longhand:**
      - margin-top: 20px; margin-right: 10px; margin-bottom: 10px; margin-left: 20px;

Đồ họa và animation trong CSS

- **CSS shadow:**
  - Chức năng: Sử dụng để tạo đổ bóng cho phần tử
  - Cách khai báo: box-shadow: value;
  - Value:
    - None: phần tử không có đổ bóng
    - h-offset: vị trí của bóng đổ theo chiều ngang
    - v-offset: vị trí của bóng đổ theo chiều dọc
    - blur: độ nhòe của bóng đổ (px)
    - spead: độ lớn của bóng đổ (px)
    - inset: chuyển đổ bóng vào bên trong phần tử
    - Inherit: phần tử được thiết lập inherit sẽ kế thừa giá trị box-shadow của phần tử cha
  - Ví dụ: box-shadow: 10px 10px 20px 20px grey;

Đồ họa và animation trong CSS…

- **CSS gradient**:
  - Chức năng: tạo màu tô chuyển sắc cho phần tử
  - Value:
    - Linear-gradient(hướng, màu 1, màu 2,…)
  - Hướng: to-top, to-bottom, deg (độ)
    - Radial-gradient(hình khối, màu 1, màu 2,...)
  - Hình khối: circle, ellipse
  - Ví dụ:
    - background-image: linear-gradient(to bottom,blue,red);
    - background-image: radial-gradient(circle,blue,red);

Đồ họa và animation trong CSS…

- **CSS transition:**
  - Chức năng: sử dụng để chuyển đổi trạng thái hiển thị của phần tử trên trang web
  - Cú pháp:
    - transition: property duration timing-function delay;
  - Trong đó:
    - *Property*: tên thuộc tính muốn áp dụng chuyển đổi
    - *Duration*: thời gian hiệu ứng chuyển đổi diễn ra (s,ms)
    - *Timing-function*: kiểu hiệu ứng chuyển đổi
    - *Delay*: thời gian chờ trước khi hiệu ứng bắt đầu

Đồ họa và animation trong CSS…

- **CSS transform**:
  - Chức năng: sử dụng để thay đổi vị trí, kích thước, hình dạng phần tử hiển thị trên trang web
  - Cú pháp:
    - transform: none|transform-functions|initial|inherit;
  - Trong đó:
    - *none*: không áp dụng hiệu ứng
    - *transform-functions*: kiểu biến đổi
    - *initial*:set các thiết lập hiện tại thành chế độ mặc định của hiệu ứng
    - *inherit*: thiết lập kế thừa giá trị của phần tử cha

Đồ họa và animation trong CSS…

- **CSS animation**:
  - Chức năng: sử dụng để tạo hoạt họa cho phần tử HTML
  - Gồm các thuộc tính:
    - @keyframes: thiết lập kiểu hoạt họa cho phần tử để hoạt họa sẽ thay đổi từ kiểu này sang kiểu khác
    - *animation-name:* tên của animation
    - *animation-duration:* thời gian hoạt họa diễn ra *animation-delay:* thời gian chờ trước khi hoạt họa bắt đầu *animation-iteration-count:* xác định số lần lặp lại của hoạt họa (*infinite*: lặp lại vô hạn lần)
    - *animation-direction:* có 2 giá trị
      - *normal*: chuyển động bình thường
      - *alternate*: chuyển động đảo ngược ở chu kì tiếp theo
    - *animation-timing-function: linear/ease/ease-in/easeout/ease-in-out/cubic-Bezier (tương tự transition)*
  - Ví dụ:

```
@keyframes change-bg-color {
  from {background-color: red;}
 to {background-color: yellow;}
}
```

3. JavaScript

Giới thiệu JavaScript

- Với HTML ta chỉ thiết kế được trang web để hiển thị thông tin, không tạo ra được sự tương tác từ phía người dùng.
- JavaScript là ngôn ngữ kịch bản (do hãng Sun từ ngôn ngữ LiveScripts) dùng để tạo các client-side scripts và serverside scripts (có sự tương tác với người dùng).
- Mặc dù có những điểm tương đồng giữa Java và JavaScipt, nhưng chúng vẫn là hai ngôn ngữ riêng biệt
- *Lưu ý*: trong code javascript cũng phân biệt chữ hoa và chữ thường

Giới thiệu JavaScript…

- JavaScript có thể chèn vào một tài liệu HTML theo những cách sau:
  - Sử dụng thẻ <SCRIPT>.
  - Sử dụng một file JavaScript từ bên ngoài.
  - Sử dụng các biểu thức JavaScript trong các giá trị thuộc tính của thẻ
  - Sử dụng JavaScript trong các trình điều khiển sự kiện

Giới thiệu JavaScript…

- **Sử dụng SCRIPT**
  - Khi trình duyệt gặp phải một thẻ <script> nào đó, nó sẽ đọc từng dòng một cho đến khi gặp thẻ đóng </script >.
  - Tiếp đến nó sẽ kiểm tra lỗi trong các câu lệnh Javascript.
  - Nếu gặp phải lỗi, nó sẽ cho hiển thị lỗi đó trong chuỗi các hộp cảnh báo (alert boxes) lên màn hình.
  - Nếu không có lỗi, các câu lệnh sẽ được biên dịch sao cho máy tính có thể hiểu được lệnh đó.

Giới thiệu JavaScript…

- **Sử dụng SCRIPT…**
  - Cú pháp:

```
<script language = "JavaScript">
// Dòng ghi chú;
/*
Đoạn ghi chú
*/
….
</script>
```

  - Việc khai báo sử dụng ngôn ngữ Javascript có thể đặt bất kỳ vị trí nào trong cặp thẻ **<html>…</html>**. Tuy nhiên, nên đặt các câu lệnh script trong phần <head> ….. </head> để đảm bảo tất cả các câu lệnh đều được đọc và biên dịch trước khi nó được gọi từ trong phần <body> … </body>

Giới thiệu JavaScript…

- **Sử dụng SCRIPT…**
  - Ví dụ:

```
<html>
<head> <title>My first page</title>
<SCRIPT LANGUAGE="JavaScript">
document.write("Xin chao cac ban!");
</SCRIPT>
</head>
<body>
<p> Nhúng  JavaScript vào trang HTML</p>
<body>
</html>
```

Giới thiệu JavaScript…

- **Sử dụng file JavaScript từ bên ngoài**
  - **File javascript** là file văn bản chứa các mã lệnh JavaScript, file javascript có phần mở rộng là **“.js”**.
  - Nó chỉ có thể chứa các câu lệnh và các hàm JavaScript, ***không thể chứa các thẻ HTML***
  - Cú pháp:

```
<script language = "JavaScript" src=”filename.js”>
</script>
```

Giới thiệu JavaScript…

- **Sử dụng file JavaScript từ bên ngoài…**
  - Ví dụ:

```
//File nhung_javascript.html:
<html>
<head> <title>My first page</title>
<SCRIPT LANGUAGE="JavaScript" src="vidu.js" > </SCRIPT>
</head>
<body>
<p> Sử dụng một file JavaScript ở ngoài</p>
<body>
</html>
```

File vidu.js:

```
document.write ("Hello! ")
```

Kiểu dữ liệu, hằng và biến

- Kiểu dữ liệu: Việc xác định kiểu dữ liệu trong javascript được chuyển đổi một cách tự động trong quá trình khai báo và sử dụng các biến. Các kiểu dữ liệu thường dùng:
  - Kiểu số nguyên
  - Kiểu số thực
  - Kiểu ký tự
  - Kiểu chuỗi
  - Kiểu logic (True – False)
  - …

Kiểu dữ liệu, hằng và biến…

- Hằng và biến:
  - Trong javascript không cho phép định nghĩa hằng tường minh
  - Các biến trong javascript phân biệt chữ hoa và chữ thường.
  - Cú pháp: **var <tên_biến>**

```
<head>
<title>Khai báo biến</title>
<script language="JavaScript">
var a; var b=2; // biến Global
var result=0; // biến Global
result=a+b; // biến Global
document.write("Ket Qua cua ham myFunction1 la : "+result+"<br>");
</script>
</head>
```

Lệnh **document.write**: dùng để xuất thông tin trên trình duyệt. (tương tự: **document.writeln**)

Các phép toán

- Trong javascript sử dụng cả hai toán tử một ngôi và hai ngôi, gồm:
  - Toán tử số học: + , - , * , / , ++ , -- , …
  - Toán tử so sánh: > , < , >= , != , …
  - Toán tử logic: AND (&&), OR (||) , …
  - Toán tử chuỗi: + (nối chuỗi)
  - Toán tử lượng giá: điều kiện (?), typeof, …
- Javascript cung cấp các thư viện hàm cho người dùng khá đầy đủ như: các hàm chuyển đổi kiểu dữ liệu, các hàm xử lý chuỗi,…

Các đối tượng hộp thoại

- Trong JavaScript cung cấp sẵn các đối tượng hộp thoại (**dialog boxes**) cho người dùng tương tác với trình duyệt trên phía client, bao gồm
  - **Alert**: hiển thị thông báo
  - **Confirm**: xác nhận thông tin người dùng
  - **Prompt**: tương tác với người dùng bằng cách cho phép nhập giá trị mới

Các đối tượng hộp thoại…

- **Alert**:
  - **Công dụng**: dùng hiển thị thông báo cho người dùng.
  - **Cú pháp**: **Alert**("Nội dung thông báo")
- **Confirm**
  - **Công dụng**: dùng xác nhận lại thông tin từ phía người dùng. Hộp thoại trả về **True** nếu người dùng đồng ý.
  - **Cú pháp**: **Confirm**("Nội dung xác nhận")
- **Prompt**
  - **Công dụng**: dùng nhận thông tin từ phía người dùng. Hộp thoại trả về **giá trị** người dùng đã nhập.
  - **Cú pháp**: **Prompt**("Tiêu đề yêu cầu nội dung")

Các đối tượng hộp thoại...

- Ví dụ:

```
<html>
<head>
<title>Dialog boxes</title>
<script language="javascript">
alert("Chào bạn");
var namsinh=prompt("Bạn sinh năm mấy?");
var traloi=confirm("Bạn có muốn tính tuổi của bạn không?");
if (traloi==true)
    document.write("<h1>Bạn được "+(2011-namsinh)+"<h1>");
else
    document.write("<h1>Chào bạn<h1>");
</script>
</head>
<body>
....
</body>
</html>
```

Các cấu trúc điều khiển cơ bản

- Cấu trúc điều kiện
  - If, If-else
  - Switch-case
- Cấu trúc lặp
  - WHILE
  - DO…WHILE
  - FOR

Mảng và hàm

- **Mảng**:
  - Trong javascript không có kiểu dữ liệu mảng ***tường minh*** **(**ví dụ: ~~int mang[10]~~**)** mà chỉ hỗ trợ thông qua đối tượng **Array** sẵn có và các thuộc tính và phương thức mà đối tượng này hỗ trợ
  - Khai báo:
    - arrayObjectName = new Array(element0,element1,..). //Hoặc
    - arrayObjectName = new Array(arrayLength)
  - Ví dụ:

```
//tạo mảng gồm 5 phần tử
var Mang = new Array(5)
```

Mảng và hàm...

- **Mảng**:...
  - **Truy cập phần tử mảng**: chỉ số bắt đầu của mảng là 0

```
<script language="javascript">
var arrMaVung = new Array("08","04","72","65","64");
var arrTenVung = new Array(5); arrTenVung[0]="HCM";
arrTenVung[1]="Hà Nội"; arrTenVung[2]="Long An";
arrTenVung[3]="Bình Dương"; arrTenVung[4]="VũngTàu";
for (i=0;i<5;i++)
    document.write(arrMaVung[i]+" "+arrTenVung[i]+"<br>");
</script>
```

Mảng và hàm...

- **Mảng:...**
  - Thuộc tính của đối tượng Array

length	Trả về số phần tử của mảng
index	Trả về chỉ mục của phần tử

  - Phương thức của đối tượng Array

oncat	Nối hai mảng và trả về một mảng mới.
join	Kết hợp tất cả các phần tử của một mảng thành một chuỗi.
Pop/shift	Gỡ bỏ phần tử cuối/đầu của một mảng và trả về phần tử đó.
Push/ unshift	Thêm một hoặc nhiều phần tử vào cuối/đầu một mảng và trả về độ dài mới của mảng.
splice	Chèn hoặc xoá một hoặc nhiều phần tử theo vị trí thứ k trong mảng
reverse	Hoán vị các phần tử của một mảng: Phần tử mảng đầu tiên trở thành phần tử cuối cùng và ngược lại
sort	Sắp xếp các phần tử của một mảng

Mảng và hàm...

- **Hàm**:
  - Javascript cung cấp sẵn cho người dùng một số hàm thông dụng:
    - Hàm **eval:** đánh giá các biểu thức hay lệnh.
    - Hàm **isFinite**: xác định xem 1 số có là hữu hạn hay không?
    - Hàm **isNaN:** kiểm tra một biến có là số?
    - Hàm **parseInt** và **parseFloat:** chuyển đổi kiểu
    - Hàm **Number** và **String:** chuyển đổi kiểu

Mảng và hàm…

- **Hàm**:…
  - Ngoài ra, người dùng có thể định nghĩa các hàm tự tạo theo cú pháp sau

```
function functionName (argument1, argument2, …) {
    statements;
    [return value;]
}
```

  - *Trong đó*:
    - function: từ khoá bắt buộc do javascript là ngôn ngữ có tính định kiểu thấp nên không cần xác định trước kiểu dữ liệu trả về.
    - **functionName**: tên hàm
    - **argument1, argument2, …**: tham số đầu vào
    - **[return value;]**: giá trị trả về của hàm nếu có

Các đối tượng cơ bản

- **Đối tượng** là kiểu dữ liệu đặc biệt và được sử dụng phổ biến trong javascript
- **Ví dụ**: đối tượng SINHVIEN có các thuộc tính: mã sinh viên, họ tên, điểm, … và các phương thức: đi học, đi thi, …
- Truy cập thuộc tính và phương thức:
  - **Tên_đối_tượng.Tên_thuộc_tính**
  - **Tên_đối_tượng.Tên_phương_thức(…)**
- Sử dụng con trỏ **This** cho đối tượng hiện hành

Các đối tượng cơ bản…

- Các **đối tượng** javascript hỗ trợ

Các đối tượng cơ bản…

- **Đối tượng cơ bản**:
  - **Array**: đối tượng dùng quản lí danh sách mảng.
  - **Math**: đối tượng liên quan đến các phép tính toán học.
  - **String**: đối tượng dùng để thao các với các chuỗi văn bản.
  - **Date**: đối tượng liên quan đến ngày giờ.

Các đối tượng cơ bản…

- **Đối tượng trình duyệt:**
  - **window**
    - **Công dụng:** dùng quản lý thông tin của tất cả các đối tượng trong cửa sổ trình duyệt.
    - **Các thuộc tính cơ bản:** đối tượng window được xem là đối tượng cơ bản (base class) của tất cả các đối tượng khác.
    - **Các phương thức cơ bản:** alert(), blur(), close(), open(), focus(), navigate().

Các đối tượng cơ bản…

- **Đối tượng trình duyệt:**
  - **Document**
    - **Công dụng:** dùng quản lý thông tin tài liệu HTML trong cửa sổ trình duyệt (được xem là đối tượng con của **window**)
    - **Các thuộc tính và phương thức:**

Các thuộc tính:

- Links: chứa danh sách tất cả các liên kết trong trang
- Form:

text

textarea

checkbox

radio

select (option)

…

write("string")	viết chuỗi đã cho trên doucment.
writeln("string")	viết chuỗi đã cho trên doucment với ký tự newline ở cuối.
getElementById()	trả về phần tử có giá trị id đã cho.
getElementsByName()	trả về tất cả các phần tử có giá trị name đã cho.
getElementsByTagName()	trả về tất cả các phần tử có tên thẻ đã cho.

Các đối tượng cơ bản…

- **Đối tượng trình duyệt:**
  - **History**
    - **Công dụng:** dùng quản lý danh sách các URL đã duyệt.
    - **Các thuộc tính và phương thức:**

Các thuộc tính:

- length
- next
- previous

Các phương thức:

- back()
- forward()

Các đối tượng cơ bản…

- **Đối tượng trình duyệt:**
  - **Location**
    - **Công dụng:** dùng quản lý thông tin URL hiện tại.
    - **Các thuộc tính và phương thức:**

Các thuộc tính:
- Hash: trả về phần sau kí hiệu # trong URL hiện tại
- Host: trả về tên miền và số cổng
- Hostname: trả về tên miền
- Href: trả về URL đầy đủ

Các phương thức:
- reload(): tải lại trang
- replace(URL): thay thế URL hiện tại bằng URL mới và chuyển hướng đến URL mới

Xử lý sự kiện

- Javascript quản lý sự tương tác giữa người dùng và trình duyệt thông qua bộ quản lý các **sự kiện (Event)** trên Form và các đối tượng con trên Form.
- **Sự kiện**: (Event): là kết quả thao tác của người dùng tác động lên đối tượng.
  - Một sự kiện bao gồm 2 thông tin
    - Kiểu sự kiện: click, double click, change,..
    - Vị trí của con trỏ tại thời điểm xảy ra sự kiện.
  - Các sự kiện thường gặp

- onClick
- onChange
- onFocus
- onBlur
- onSelect
- onMouseOver

- onMouseOut
- onLoad
- onUnload
- onSubmit
- onMouseDown
- onMouseUp

Xử lý sự kiện...

- **Các sự kiện cơ bản trên các đối tượng:**
  - **Đối tượng window: (onLoad - onUnload)**
    - **Công dụng**: sự kiện xảy ra khi người dùng mở trang hoặc đóng trang.
    - **Ví dụ**: viết trang khi người dùng mở trang thì hiển thị thông báo nhập Tên -> xuất "Hi, welcome my pages", sau khi đóng trang thì hiển thị thông báo "Good bye, see you again !"

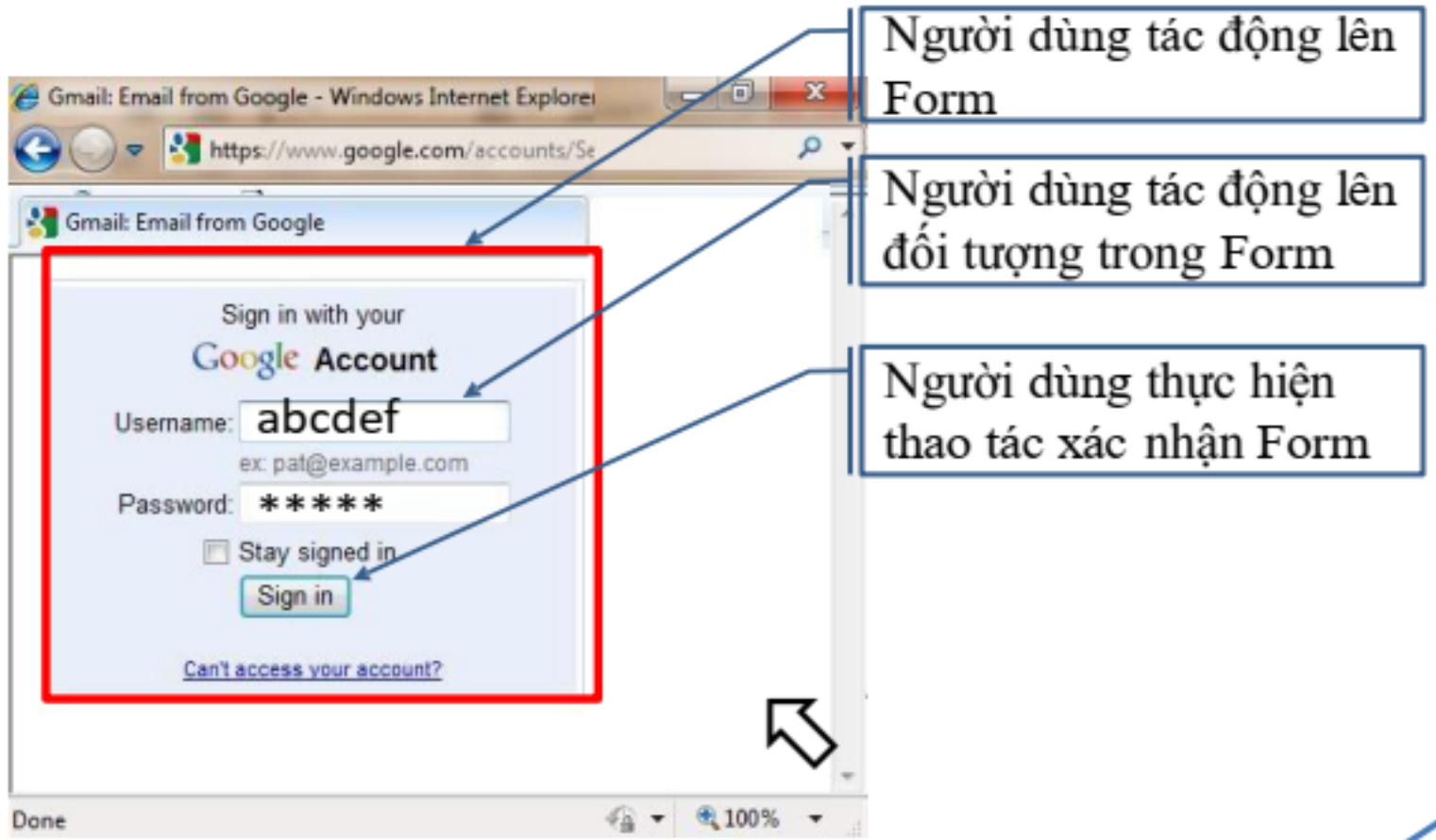
```
<html>
<head>
</head>
<body onload="hello();" onunload="goodbye();">
</body>
</html>
```

```
<script language="javascript">
        var name = "";
        function hello() {
                name = prompt('Enter Your Name:','Name');
                alert('Hi ' + name + ', welcome to my page!');
        }
        function goodbye() {
                alert('Goodbye ' + name + ', see you again!');
        }
</script>
```

Xử lý sự kiện...

- **Các sự kiện cơ bản trên các đối tượng:**
  - **Đối tượng form**
    - Sự kiện trên form được xử lý phụ thuộc vào 2 yếu tố sau:
      - **Thuộc tính sự kiện** của form: **Action, Method, …**
      - Việc xử lý các sự kiện của các đối tượng con (button, textbox,..) bên trong form: **onSubmid, onClick, onBlur, onChange, …**
    - Ví dụ về chu trình sự kiện trên form:

Xử lý sự kiện...



Người dùng tác động lên Form

Người dùng tác động lên đối tượng trong Form

Người dùng thực hiện thao tác xác nhận Form

Bộ lắng nghe sự kiện Javascript hoạt động.

OnMouseOver → OnChange (tuỳ đối tượng) → OnClick OnSubmit → Xử lý Form phụ thuộc vào các thuộc tính sự kiện

Xử lý sự kiện...

- **Các sự kiện cơ bản trên các đối tượng:**
  - **Đối tượng form...**
    - **Thuộc tính sự kiện:**
      - **Action**: thuộc tính dùng để chuyển trang hiện hành đến một trang khác theo địa chỉ URL truyền vào.
      - **Method**: phương thức gửi nội dung đi: “get / post”