**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

*Trà Vinh, ngày ….. tháng …… năm ……*

**Giáo viên hướng dẫn**

*(Ký tên và ghi rõ họ tên)*

*Trà Vinh, ngày ….. tháng …… năm ……*

**Giáo viên hướng dẫn**

*(Ký tên và ghi rõ họ tên)*

**LỜI CẢM ƠN**

**NHẬN XÉT CỦA THÀNH VIÊN HỘI ĐỒNG**

*Trà Vinh, ngày ….. tháng …… năm ……*

**Thành viên hội đồng**

*(Ký tên và ghi rõ họ tên)*

MỤC LỤC

DANH MỤC HÌNH ẢNH – BẢNG BIỂU

# TÓM TẮT

# MỞ ĐẦU

# CHƯƠNG 1: TÌM HIỂU HTML, CSS VÀ JAVASCRIPT

## Ngôn ngữ HTML

### Tổng quan HTML

HTML (HyperText Markup Language hay "Ngôn ngữ Đánh dấu Siêu văn bản") là một [ngôn ngữ đánh dấu](https://vi.wikipedia.org/wiki/Ng%C3%B4n_ng%E1%BB%AF_%C4%91%C3%A1nh_d%E1%BA%A5u) được thiết kế ra để tạo nên các [trang web](https://vi.wikipedia.org/wiki/Website) với các mẩu thông tin được trình bày trên [World Wide Web](https://vi.wikipedia.org/wiki/World_Wide_Web). Cùng với [CSS](https://vi.wikipedia.org/wiki/CSS) và [JavaScript](https://vi.wikipedia.org/wiki/JavaScript), HTML tạo ra bộ ba nền tảng kỹ thuật cho [World Wide Web](https://vi.wikipedia.org/wiki/World_Wide_Web). HTML đã trở thành một chuẩn [Internet](https://vi.wikipedia.org/wiki/Internet) do tổ chức [World Wide Web Consortium](https://vi.wikipedia.org/wiki/W3C) (W3C) duy trì. Phiên bản chính thức mới nhất của HTML là HTML5.

HTML mô tả cấu trúc của một trang web. HTML bao gồm một tập hợp các phần tử. Các phần tử sẽ chỉ định cách hiển thị nội dung trên trình duyệt. Các phần tử được diễn tả bởi các thẻ (tag). Trình duyệt không hiển thị các thẻ, nhưng sử dụng chúng để hiển thị nội dung của trang web.

#### Trình duyệt web

Trình duyệt web hay Web Browser là một phần mềm ứng dụng được gọi tắt là trình duyệt, trình duyệt web giúp người dùng có thể nhìn thấy và tương tác với các văn bản, hình ảnh, nhạc, video, các trò chơi và nhiều thông tin khác xuất hiện trên một trang web bất kì.

Mỗi trình duyệt sẽ có một cách đọc và hiển thị HTML riêng, nên có thể là cùng một trang web nhưng khi truy cập bằng hai trình duyệt khác nhau ta sẽ thấy hai cách hiển thị khác nhau trên hai trình duyệt.

Các trình duyệt web phổ biến hiện nay như: Google Chrome, Cốc Cốc, Firefox, Safari, Internet Explorer...

#### Trình soạn thảo mã nguồn

Trình soạn thảo mã nguồn hay Code Editor là công cụ không thể thiếu trong việc thiết kế xây dựng website. Trình soạn thảo mã nguồn giúp chúng ta làm nổi bật code (Highlight code), tự động hoàn thiện (Auto complete), sửa lỗi (Debug) và các tính năng khác giúp cho việc soạn thảo mã nguồn được tiện lợi hơn, nhanh chóng hơn và dễ dàng tìm kiếm, sửa lỗi.

Một số trình soạn thảo hỗ trợ việc soạn thảo mã nguồn:

* Notepad
* Notepad++
* Sublime Text
* Visual Studio Code
* TextMate
* Komodo Edit

#### Client – Server

Mô hình client-server là một mô hình nổi tiếng trong mạng máy tính, được áp dụng rất rộng rãi và là mô hình của mọi trang web hiện có.  Ý tưởng của mô hình này là máy con - client (đóng vài trò là máy khách) gửi một yêu cầu (request) để máy chủ - server (đóng vai trò người cung ứng dịch vụ), máy chủ sẽ xử lý và trả kết quả về cho máy khách.

Ưu điểm của mô hình client-server là giúp tối ưu hóa việc lưu trữ và chia sẻ dữ liệu bằng cách tập trung chúng vào một máy chủ duy nhất thay vì việc phân tán cùng một nội dung trên nhiều máy khách khác nhau.

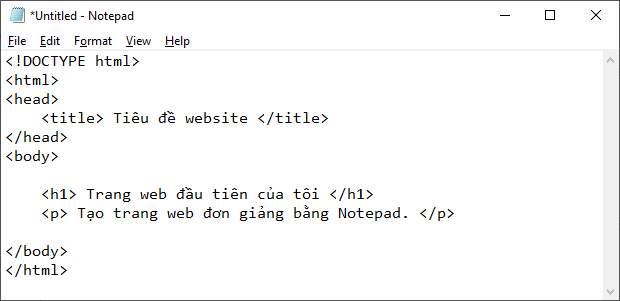
Nhược điểm của mô hình này là tính an toàn và bảo mật các thông tin trên mạng. Do phải trao đổi các dữ liệu giữa hai máy ở hai khu vực khác nhau nên dễ xảy ra hiện tượng thông tin bị lộ và phát tán khắp nơi.

### Tạo một trang web đơn giản

Để tạo một trang web đơn giản, ta thực hiện các bước sau:

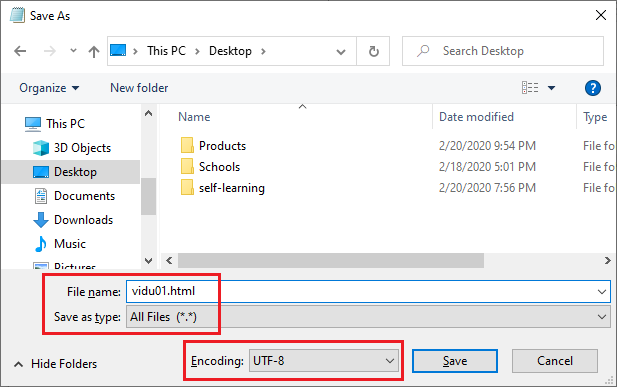
Bước 1: Ta sẽ sử dụng trình soạn thảo Notepad trên máy tính Window để xây dựng trang web. Vào **Menu Start** > gõ **Notepad** > nhấn **Enter**.

Bước 2: Nhập các dòng code như hình sau.



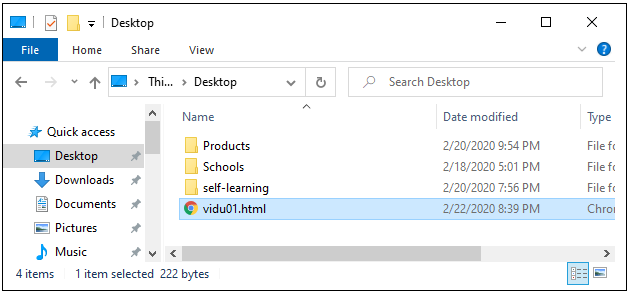
Hình : Soạn thảo mã nguồn tạo trang web đơn giản bằng Notepad.

Bước 3: Chọn **File > Save** lưu lại tập tin với phần mở rộng “**.html**” hoặc “**.htm**” và chọn **Encoding** là **UTF-8** như hình.



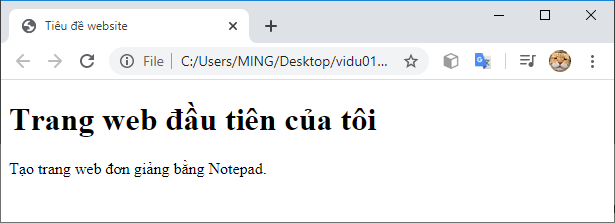
Hình : Lưu tập tin với định dạng html và UTF-8.

Bước 4: Kiểm tra xem tập tin đã được lưu đúng định dạng hay chưa. Nếu biểu tượng tập tin là một trình duyệt bất kì có trên máy tính thì chúng ta đã tạo thành công một trang web đơn giản.



Hình : Kiểm tra tập tin có được lưu đúng định dạng hay không.

Bước 5: Xem trang web bằng cách nhấn đúp chuột vào tập tin html và mở bằng trình duyệt mà chúng ta muốn.



Hình : Xem trang web trên trình duyệt.

### Cấu trúc trang HTML

<!DOCTYPE html>  
<html>  
<head>  
 <title> Page Title </title>  
</head>  
<body>  
 <p> Element HTML </p>  
</body>  
</html>

Trong đó:

* *<!DOCTYPE html>* là phần khai báo phiên bản HTML.
* *<html>* là thẻ bao bọc nội dung của trang HTML.
* *<head>* là thẻ khai báo thông tin của trang web.
* *<title>* là thẻ dùng để khai báo tiêu đề của trang web.
* *<body>* là thẻ chứa nội dung trang và được hiển thị trên trình duyệt.
* *<p>* là thẻ khai báo đoạn văn bản.

### Các thẻ HTML cơ bản

Thẻ (tag) hay còn gọi là phần tử trong HTML có hai loại: thẻ kép và thẻ đơn.

* Thẻ kép là loại có thẻ mở và thẻ đóng, ví dụ: *<tên\_thẻ> ... </tên\_thẻ>*. Nội dung cần hiển thị sẽ được đặt giữa cặp thẻ mở và thẻ đóng.
* Thẻ đơn là loại thẻ chỉ có thẻ mở mà không có thẻ đóng, ví dụ: *<tên\_thẻ>*.

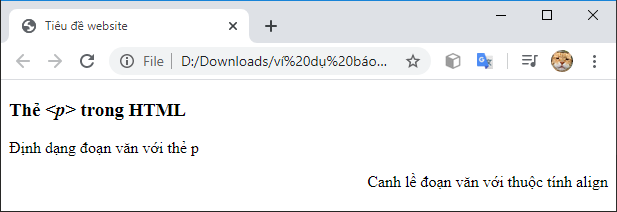
#### Thẻ đoạn văn

Trong HTML, thẻ *<p>* được dùng để định dạng đoạn văn bản. Khi kết thúc đoạn văn được đánh dấu bằng thẻ *</p>*. Nếu muốn canh lề đoạn văn, ta có thể sử dụng thuộc tính *align* với bốn giá trị: *center / justify / left / right.*

<p> Định dạng đoạn văn với thẻ p </p>

<p align="right"> Canh lề đoạn văn với thuộc tính align </p>

Kết quả khi hiển thị trên trình duyệt:



Hình : Định dạng các đoạn văn với thẻ <p>.

#### Thẻ tiêu đề

HTML định nghĩa sẵn sáu cấp độ của tiêu đề: <*h1>…</h1>* cho đến *<h6>…</h6>* với độ lớn giảm dần từ *h1* đến *h6*. Kết thúc thẻ tiêu đề, nội dung trình bày cũng tự động xuống dòng.

<h1>Tiêu đề 1</h1>

<h2>Tiêu đề 2</h2>

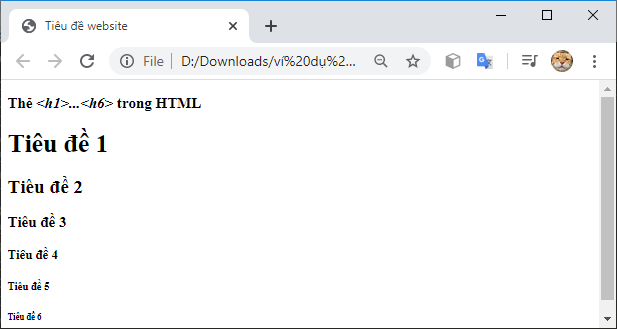
<h3>Tiêu đề 3</h3>

<h4>Tiêu đề 4</h4>

<h5>Tiêu đề 5</h5>

<h6>Tiêu đề 6</h6>

Kết quả khi hiển thị trên trình duyệt:



Hình : Sử dụng thẻ <h1> đến <h6> để định dạng tiêu đề.

#### Thẻ định dạng chữ

Có rất nhiều thẻ định dạng chữ tùy theo ý định của người thiết kế Web mà vận dụng các thẻ này một cách linh hoạt.

Khi hiển thị nội dung, phím *Enter* sẽ không có tác dụng. Vì vậy ta có thể dùng thẻ *<br>* để ngắt dòng. Đây là một thẻ đơn và không có thuộc tính bên trong.

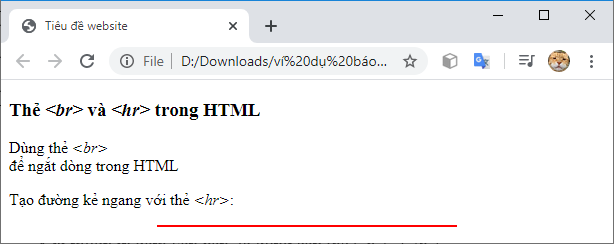
Ta có thể sử dụng thẻ *<hr>* để tạo một đường kẻ ngang trên trang web, đây cũng là một thẻ đơn nhưng có ba thuộc tính: *color* dùng để chỉ định màu sắc cho đường kẻ, *width* dùng để chỉ định độ dài của đường kẻ và cuối cùng là thuộc tính *size* dùng để chỉ định độ dày của đường kẻ.

<p> Dùng thẻ <br> để ngắt dòng trong HTML </p>

<p> Tạo đường kẻ ngang với thẻ <hr>: </p>

<hr color="red" width="50%" size="5">

Kết quả khi hiển thị trên trình duyệt:



Hình : Cách dùng thẻ <br> và <hr> trong HTML.

Khi muốn in đậm văn bản, ta dùng cặp thẻ *<b>...</b>*.

Khi muốn in nghiêng văn bản, ta dùng cặp thẻ *<i>...</i>*.

Khi muốn gạch chân văn bản, ta dùng cặp thẻ *<u>...</u>*.

Khi muốn nhấn mạnh văn bản, ta dùng cặp thẻ *<strong>...</strong>*. Nội dung văn bản sẽ được in đậm như thẻ *<b>*.

Khi muốn làm nổi bật văn bản, ta dùng cặp thẻ *<em>...</em>*. Nội dung văn bản sẽ được in nghiêng như thẻ *<i>*.

Để giữ nguyên định dạng văn bản như trong trình soạn thảo, ta dùng cặp thẻ *<pre>...</pre>*.

Trường hợp ta muốn biểu diễn một phương trình toán học hay hóa học ta dùng cặp thẻ *<sup>...</sup>* để biểu diễn số mũ và cặp thẻ *<sub>...</sub>* để biểu diễn chỉ số dưới.

<p align="left"> <b> Đoạn văn bản in đậm </b> </p>

<p align="center"> <i> Đoạn văn bản in nghiêng </i> </p>

<p align="right"> <u> Đoạn văn bản được gạch chân </u> </p>

<p> Phương trình toán học: ax<sup>2</sup> + bx + c = 0 </p>

<p> Kí hiệu hóa học: H<sub>2</sub>O </p>

<pre>

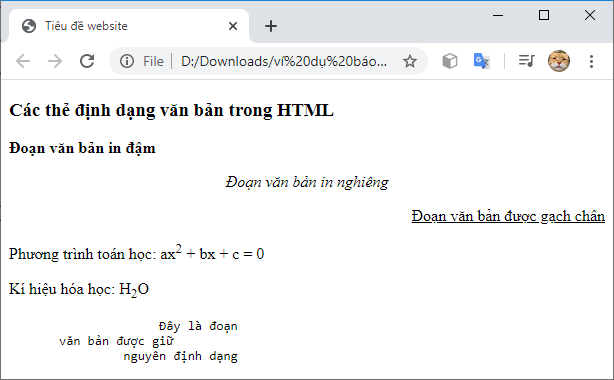
Đây là đoạn

văn bản được giữ

nguyên định dạng

</pre>

Kết quả khi hiển thị trên trình duyệt:



Hình : Các thẻ định dạng chữ cơ bản được dùng trong HTML.

#### Thẻ định dạng danh sách

Cũng như những trình soạn thảo khác, trang web cũng có thể hiển thị nội dung theo dạng danh sách. Trong HTML, có hỗ trợ hai loại danh sách đó là danh sách có thứ tự *<ol></ol>* và danh sách không thứ tự *<ul></ul>*. Các mục nội dung sẽ tương ứng với các thẻ *<li></li>* bên trong hai thẻ danh sách trên.

Đối với dạng danh sách có thứ tự, HTML hỗ trợ thuộc tính *type* với các định dạng có sẵn như dạng số (1, 2, 3,...), ký tự chữ thường (a, b, c,...), ký tự chữ hoa (A, B, C,...), số La Mã thường (i, ii, iii,...), số La Mã hoa (I, II, III,...).

Thuộc tính *start* sẽ chỉ định thứ tự bắt đầu của danh sách. Trong trường hợp không gọi được thuộc tính *start* thì mặc định *start = 1*.

<ol type="1/a/A/i/I" start="n">

<li> Mục thứ 1 </li>

<li> Mục thứ 2 </li>

...

<li> Mục thứ n </li>

</ol>

Đối với dạng danh sách không thứ tự, HTML hỗ trợ thuộc tính *type* với các định dạng có sẵn như hình tròn (*circle*), chấm tròn (*disc*), chấm vuông (*square*) và không có thuộc tính *start.*

<ul type="circle/disc/square">

<li> Mục thứ 1 </li>

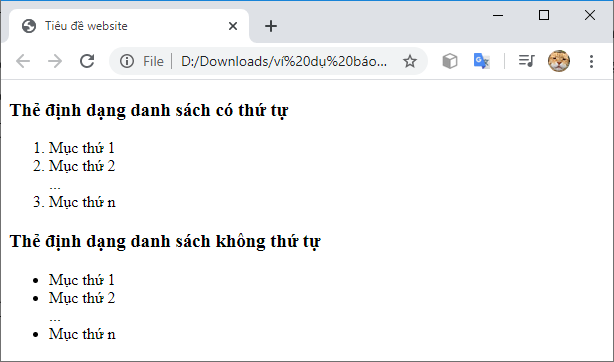
<li> Mục thứ 2 </li>

...

<li> Mục thứ n </li>

</ul>

Kết quả khi hiển thị trên trình duyệt:



Hình : Danh sách có thứ tự và không thứ tự trong HTML.

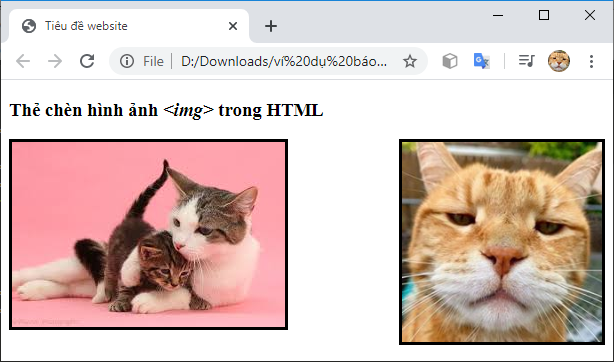
#### Thẻ chèn hình ảnh, âm thanh

Hình ảnh trên trang web được chia thành hai dạng. Dạng thứ nhất là hình ảnh làm nền, tức là chúng ta có thể hiển thị chữ trên hình ảnh. Ở dạng này, chúng ta có thể chèn hình trong thuộc tính background của một số thẻ hỗ trợ ảnh nền như *<body>*, *<table>*, *<td>*,…. Dạng thứ hai, hình ảnh được xem như một đối tượng trên trang web, nó chiếm một vị trí như một nội dung của trang, với dạng này ta dùng thẻ *<img>* để chèn hình ảnh vào, đây là một thẻ đơn với các thuộc tính hỗ trợ bên trong.

<img src="images/cat.png" alt="cat.png" border="5" align="right">

Một số thuộc tính thường đi kèm với thẻ *<img>*:

* *src*: là đường dẫn đến hình ảnh cần chèn vào trang web.
* *title*: sẽ hiển thị nội dung chú thích khi rê chuột vào hình ảnh.
* *align*: canh lề cho hình ảnh.
* *width, height*: chỉ định kích thước ảnh khi hiển thị trên trang web, nếu bỏ qua hai thuộc tính này, trình duyệt sẽ hiển thị hình ảnh với kích thước mặc định của ảnh.
* *alt*: hiển thị chú thích khi hình ảnh không hiển thị được trên trang web.
* *border*: thiết lập đường viền cho ảnh, mặc định là 0, nếu muốn thêm đường viền chỉ cần nhập một số nguyên dương vào.



Hình : Chèn hình ảnh vào trang web với thẻ <img>.

#### Thẻ liên kết

Trong website luôn tồn tại các link liên kết với nhau. Link trong HTML được chia thành hai loại, đó là link nội bộ chỉ liên kết đến những thành phần, những trang nằm trong nội bộ trang web và link đến một nơi khác trên mạng Internet như một trang web khác.

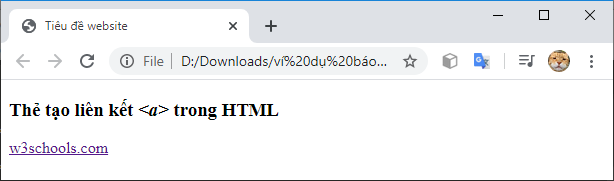
Trong HTML, sử dụng thẻ *<a>* để tạo một liên kết. Bên trong đó là các thuộc tính mà ta có thể dùng.

<a href="https://www.w3schools.com" target="\_blank" title="Tutorials web"> w3schools.com </a>

Trong đó:

* *href:* là đường dẫn đến liên kết.
* *target:* là các tùy chọn cách hiển thị kết quả liên kết.
* *title:* là tiêu đề cho đường link.

Kết quả khi hiển thị trên trình duyệt:



Hình : Thẻ <a> dùng để tạo liên kết trong trang web.

#### Thẻ tạo bảng

Khi ta muốn tạo bảng dữ liệu trên trang web, ta cần phải sử dụng thẻ *<table></table>*, bên trong cặp thẻ table ta lại dùng cặp thẻ *<tr></tr>* để tạo dòng trong bảng và sau đó là cặp thẻ *<td></td>* để tạo cột. Ngoài ra, còn có thẻ *<th></th>* dùng để khai báo tiêu đề cho bảng.

<table>

<tr>

<td> Dòng 1 Cột 1 </td>

<td> Dòng 1 Cột 2 </td>

</tr>

<tr>

<td> Dòng 2 Cột 1 </td>

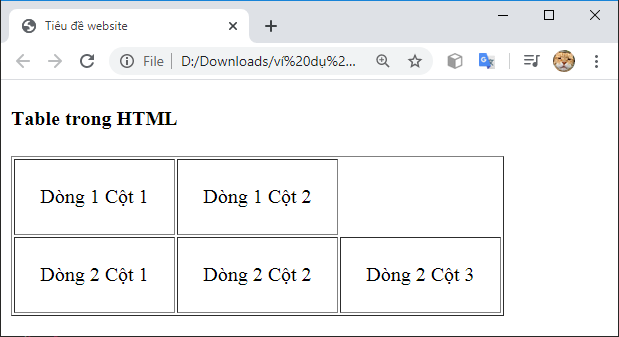
<td> Dòng 2 Cột 2 </td>

<td> Dòng 2 Cột 3 </td>

</tr>

</table>

Kết quả khi hiển thị trên trình duyệt:



Hình : Tạo bảng trình bày dữ liệu bằng thẻ <table>.

#### Form

Trang web không chỉ dừng lại ở việc hiển thị thông tin mà nó còn có thể cho phép người dùng tương tác trên nó. Ví dụ: người dùng có thể nhập thông tin lên biểu mẫu và gửi về máy chủ xử lý.

Thẻ *<form></form>* là một thẻ HTML cung cấp các biểu mẫu nhập liệu và các thành phần bên trong nó.

<form method="" action="">

<!—- Xử lý nhập liệu trong form -->

</form>

Thuộc tính *method* xác định phương thức gửi dữ liệu về máy chủ. Nếu phương là *POST* thì các thông tin nhập liệu của người dùng sẽ không hiển thị trên thanh địa chỉ của trình duyệt. Ngược lại, nếu phương thức là *GET* thì các thông tin mà người dùng gửi đi sẽ được nhìn thấy trên thanh địa chỉ của trình duyệt. Mặc định phương thức truyền dữ liệu là *GET*. Do đó, vì vấn đề bảo mật nếu trường hợp biểu mẫu nhập liệu có những thông tin cá nhân cần được giữ bí mật như mật khẩu, số tài khoản ngân hàng,...thì phải dùng phương thức *POST.*

Thuộc tính *action* sẽ trỏ đến địa chỉ mà nó được cung cấp. Đó có thể là một địa chỉ của trang sẽ nhận thông tin người dùng nhập vào để xử lý khi được gửi về máy chủ, hoặc cũng có thể để rỗng nếu muốn thông tin khi gửi đi sẽ được xử lý tại chính trang biểu mẫu này.

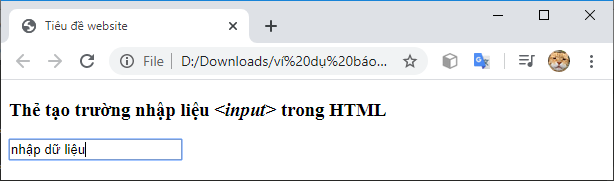
#### Thẻ tạo trường nhập liệu

Text Field hay thẻ *<input>* là một khung nhập liệu cho phép người dùng nhập nội dung vào. *<input>* là một thẻ đơn và tùy theo giá trị được gọi trong thuộc tính *type* của thẻ này mà có thể quy định các dạng nhập dữ liệu khác nhau, ví dụ: *type="text"*, *type="number"*,...

Các thuộc tính của thẻ *<input>* với các công cụng như sau:

* *maxlength*: quy định chiều dài tối đa của chuỗi nôi dung được nhập vào.
* *value*: thiết lập giá trị mặc định cho trường nhập liệu.
* *readonly*: xác định giá trị trường nhập liệu chỉ có thể đọc, không thể thay đổi.
* *disabled*: xác định trường nhập liệu bị vô hiệu hóa và không thể sử dụng.
* *size*: xác định kích thước cho trường nhập liệu.
* *type*: xác định kiểu dữ liệu được nhập vào.

<input type="text" size="20" maxlength="20">

Kết quả khi hiển thị trên trình duyệt: 

Hình : Nhập dữ liệu với thẻ <input>.

## Bảng định kiểu CSS

CSS là chữ viết tắt của Cascading Style Sheets. CSS mô tả cách thức hiển thị của các phần tử HTML trên trang web. CSS có thể điều khiển bố cục của nhiều trang web cùng lúc, nó giúp tiết kiệm công sức, thời gian để định kiểu cho website.

Giống như HTML, CSS không thực sự là ngôn ngữ lập trình. Nó cũng không phải là ngôn ngữ đánh dấu mà nó là một ngôn ngữ định kiểu. Điều này có nghĩa là nó cho phép chúng ta áp dụng kiểu có chọn lọc cho các phần từ trong tài liệu HTML.

CSS được phát triển bởi W3C (World Wide Web Consortium) vào năm 1996. Mối tương quan giữa HTML và CSS rất mật thiết. HTML là ngôn ngữ markup (nền tảng của trang web) và CSS định hình phong cách (tất cả những gì mà chúng ta thấy trên giao diện website), chúng không thể tách rời.

### Sử dụng CSS

Có 3 cách sử dụng định kiểu CSS trong HTML: định kiểu nội dòng (inline style sheet), định kiểu bên trong (internal style sheet) và định kiểu bên ngoài (external style sheet).

Các định kiểu nội dòng ít khi được sử dụng . Với định kiểu nội dòng, các quy tắc định kiểu được lồng vào bên trong các thẻ HTML, ví dụ:

<h1 style="color: red"> Tiêu đề này sẽ có màu đỏ </h1>

Các định kiểu nội dòng rất mạnh, vì chúng ghi đè tất cả các định kiểu bên trong và bên ngoài khác, mặc dù vậy chúng chỉ áp dụng định kiểu cho một thẻ duy nhất tại một thời điểm và không có khả năng tái sử dụng.

Định kiểu bên trong là dạng định kiểu được nhúng sử dụng cặp thẻ *<style></style>* và được đặt trong phần *<head>* của trang HTML. Khi sử dụng dạng định kiểu này, để có thể định kiểu cho một thẻ hay phần tử trong HTML ta chỉ việc gọi đến tên thẻ hoặc tên mà ta đặt cho thẻ để không trùng với các thẻ khác. Dạng định kiểu này có độ ưu tiên thấp hơn dạng nội dòng nhưng cao hơn dạng định kiểu bên ngoài.

<head>

<style>

Selector {

property: value;

property: value;

}

</style>

</head>

Định kiểu bên ngoài là một tập tin riêng biệt với tên mở rộng là *.css*. Khi sử dụng định kiểu bên ngoài, tất cả các quy tắc định kiểu nằm trong tập tin *.css* và cần tạo liên kết giữa nó tới trang HTML. Trong khi các định kiểu nội dòng, định kiểu bên trong chỉ tác động đến trang mà nó tồn tại thì định kiểu bên ngoài có thể định kiểu cho cùng lúc nhiều trang.

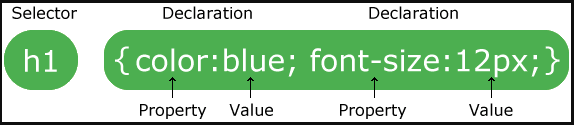
<head>

<link rel="stylesheet" href="external\_Style.css">

</head>

### Cú pháp CSS

Cú pháp CSS được thể hiện qua bộ chọn và khối khai báo.



Hình : Bộ quy tắc sử dụng CSS gồm có bộ chọn và khối khai báo.

Bộ chọn (*selector*) trỏ đến phần tử HTML cần định kiểu.

Khối khai báo (*declaration*) được đặt trong cặp dấu ngoặc nhọn {} chứa một hay nhiều khai báo cách nhau bởi dấu chấm phẩy (;).

Mỗi khối khai báo gồm tên thuộc tính CSS (*property*) và giá trị (*value*).

Ngoài bộ chọn là tên thẻ ra, ta còn có thể trỏ đến phần tử HTML thông qua thuộc tính *id* và *class.* Ví dụ sau sẽ trỏ đến phần tử HTML có *id* là *myHeading.*

#myHeading {

color: blue;

font-size: 120px;

}

Trỏ đến phần tử HTML có tên *class* là *myParagraph.*

.myParagraph {

color: gray;

font-size: 48px;

}

### Các thuộc tính định dạng văn bản trong CSS

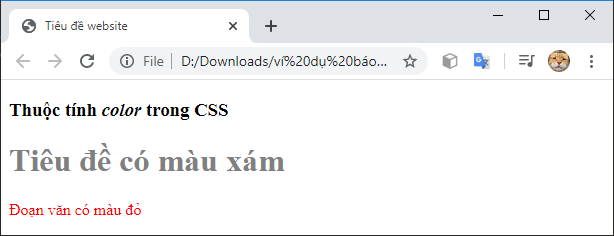
#### Color

Trong CSS ta có thể sử dụng thuộc tính *color* với giá trị là tên màu bằng tiếng anh hoặc mã màu tương ứng để thiết lập màu sắc cho văn bản. Ví dụ sau sẽ thiết lập cho các tiêu đề có màu xám và các đoạn văn có màu đỏ (mã màu tương đương của màu đỏ là #f00).

<h1 style="color: gray;"> Tiêu đề có màu xám </h1>

<p style="color: #f00;"> Đoạn văn có màu đỏ </p>

Kết quả khi hiển thị trên trình duyệt:



Hình : Thiết lập màu sắc với thuộc tính color CSS.

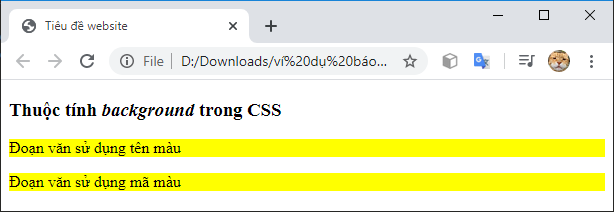
#### Background

Thuộc tính *background* dùng để thiết lập màu nền cho một đoạn văn. Cách sử dụng cũng như thuộc tính *color* là ta có thể sử dụng tên màu bằng tiếng anh hoặc mã màu tương ứng. Ví dụ, đoạn văn sau sẽ có màu nền là vàng.

<p style="background: yellow;"> Đoạn văn sử dụng tên màu </p>

<p style="background: #ffff00;"> Đoạn văn sử dụng mã màu </p>

Kết quả khi hiển thị trên trình duyệt:



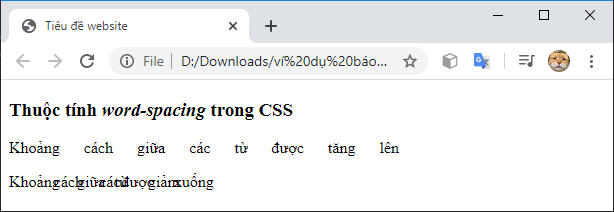
Hình : Thiết lập màu nền với thuộc tính background.

#### Word-spacing

Thuộc tính *word-spacing* dùng để thiết lập khoảng cách giữa các từ, có thể tăng hoặc giảm và có thể sử dụng giá trị âm. Giá trị khoảng cách được tính bằng pixel.

<p style="word-spacing: 20px;"> Khoảng cách giữa các từ được tăng lên </p>

Kết quả khi hiển thị trên trình duyệt:



Hình : Thiết lập khoảng cách các từ với thuộc tính word-spacing.

#### Letter-spacing

Thuộc tính *letter-spacing* dùng để thiết lập khoảng cách giữa các kí tự. Tương tự như thuộc tính *word-spacing*, thuộc tính này có thể tăng hoặc giảm khoảng cách giữa các kí tự và cũng có thể nhận giá trị âm. Giá trị khoảng cách được tính bằng pixel.

<p style="letter-spacing: 20px;"> Khoảng cách giữa các kí tự được tăng lên </p>

<p style="letter-spacing: -5px;"> Khoảng cách giữa các kí tự được giảm xuống </p>

Kết quả khi hiển thị trên trình duyệt:



Hình : Thiết lập khoảng cách giữa các kí tự với thuộc tính letter-spacing.

#### Text-align

Thuộc tính *text-align* dùng để canh lề văn bản với bốn giá trị có thể thiết lập: *center / justify / left / right*.

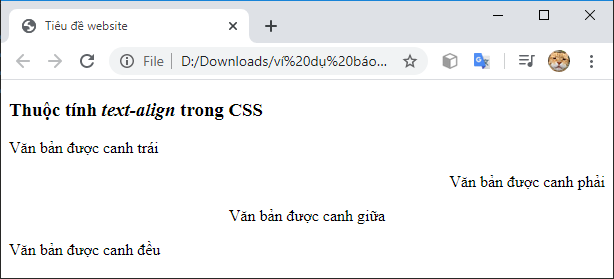
<p style="text-align: left"> Văn bản được canh trái</p>

<p style="text-align: right"> Văn bản được canh phải</p>

<p style="text-align: center"> Văn bản được canh giữa</p>

<p style="text-align: justify"> Văn bản được canh đều</p>

Kết quả khi hiển thị trên trình duyệt:



Hình : Canh lề văn bản với thuộc tính text-align.

#### Text-transform

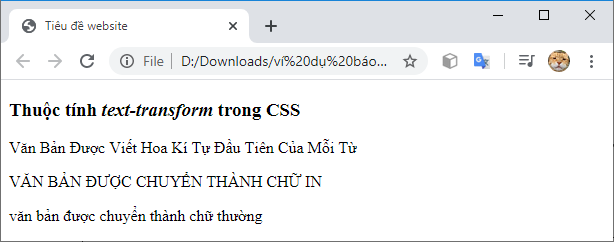
Thuộc tính *text-transform* có ba giá trị được dùng để biến đổi văn bản: *capitalize* / *uppercase /* *lowercase*.

<p style="text-transform: capitalize;"> Văn bản được viết hoa kí tự đầu tiên của mỗi từ </p>

<p style="text-transform: uppercase;"> Văn bản được chuyển thành chữ in </p>

<p style="text-transform: lowercase;"> Văn bản được chuyển thành chữ thường </p>

Kết quả khi hiển thị trên trình duyệt:



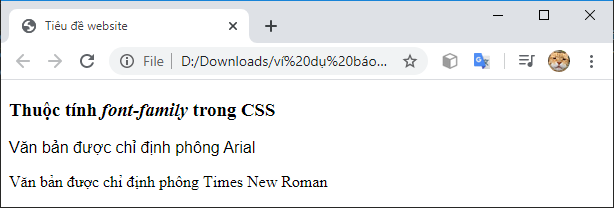
Hình : Biến đổi văn bản với thuộc tính text-transform.

#### Font-family

Thuộc tính *font-family* dùng để thiết lập phông chữ cho văn bản. Nếu máy tính người dùng không có phông được chỉ định thì trình duyệt sẽ lấy phông mặc định có cùng họ với phông được chỉ định để hiển thị.

<p style="font-family: Arial, Helvetica, sans-serif;"> Văn bản được chỉ định phông Arial</p>

Kết quả khi hiển thị trên trình duyệt:



Hình : Chỉ định phông chữ với thuộc tính font-family.

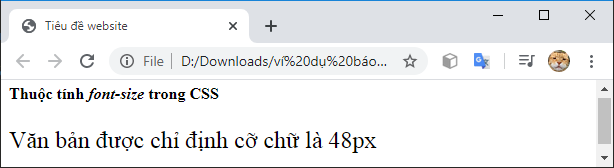
#### Font-size

Thuộc tính *font-size* được dùng để thiết lập cỡ chữ với các giá trị như: *xx-small*, *x-small*, *small*, *medium*, *large*, *x-large*, *xx-large*.

Ngoài ra, ta còn có thể chỉ định cỡ chữ cụ thể bằng đơn vị pixel, cách chỉ định này được dùng thường xuyên trong thiết kế web.

<p style="font-size: 48px;"> Văn bản được chỉ định cỡ chữ là 48px</p>

Kết quả khi hiển thị trên trình duyệt:

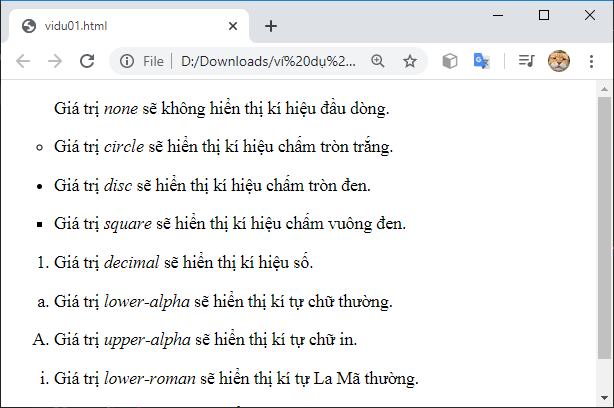


Hình : Thiết lập cỡ chữ với thuộc tính font-size.

### Thuộc tính định dạng danh sách trong CSS

Trong CSS ta có thể định kiểu cho danh sách bằng thuộc tính *list-style* với nhiểu giá trị khác nhau.

|  |  |  |
| --- | --- | --- |
| **Thuộc tính** | **Giá trị** | **Ý nghĩa** |
| *list-style* | *none* | Không dùng kí hiệu |
| *circle* | Chấm tròn rỗng ○ |
| *disc* | Chấm tròn đen ● |
| *square* | Chấm vuông đen ■ |
| *decimal* | Kí tự số (1, 2, 3,...) |
| *lower-alpha* | Kí tự chữ thường (a, b, c,...) |
| *upper-alpha* | Kí tự chữ hoa (A, B, C,...) |
| *lower-roman* | Kí tự La Mã thường (i, ii, iii,...) |
| *upper-roman* | Kí tự La Mã in (I, II, III,...) |



Hình : Định kiểu danh sách với thuộc tính list-style.

### Các thuộc tính định kiểu nền trong CSS

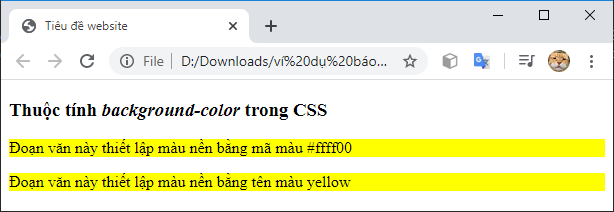
#### Background-color

Thuộc tính *background-color* dùng để thiết lập màu nền cho phần tử HTML như *body, div, p,*...Tương tự như thuộc tính *color*, ta có thể sử dụng tên màu bằng tiếng anh hoặc mã màu tương ứng. Ví dụ:

<p style="background-color: #ffff00">Đoạn văn này thiết lập màu nền bằng mã màu #ffff00 </p>

<p style="background-color: yellow ">Đoạn văn này thiết lập màu nền bằng tên màu yellow </p>

Kết quả khi hiển thị trên trình duyệt:



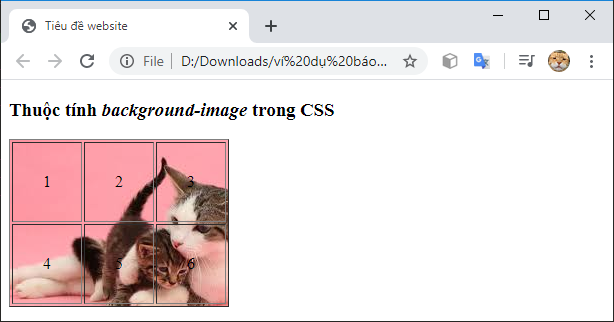
Hình : Thiết lập màu nền với thuộc tính background-color.

#### Background-image

Nếu muốn dùng hình ảnh để làm nền cho trang web, ta có thể dùng thuộc tính *backround-image*. Giá trị của thuộc tính này là một đường dẫn đến hình ảnh cần chèn vào trang web làm nền.

<div style="background-image: url('images.png'); height: 200px;"></div>

Kết quả khi hiển thị trên trình duyệt:



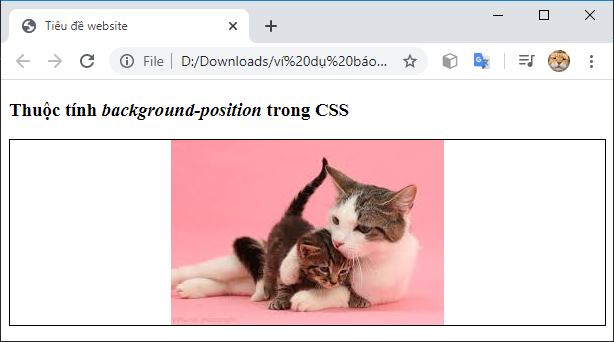
Hình : Thiết lập ảnh nền cho table với thuộc tính background-image.

#### Background-position

Thuộc tính *background-position* dùng để thiết lập vị trí đặt ảnh, với các giá trị như: *top* (trên), *bottom* (dưới), *left* (trái), *right* (phải), *center* (giữa).

<div style="background-image: url('images.png'); height: 200px; background-position: center;"></div>

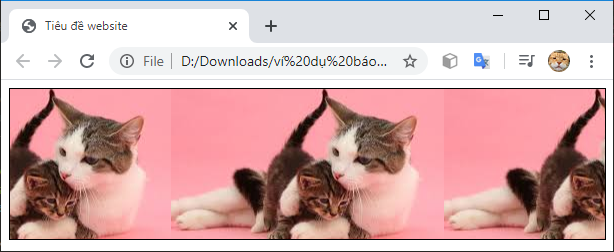
Kết quả khi hiển thị trên trình duyệt:



Hình : Định vị trí đặt ảnh với thuộc tính background-position.

#### Background-repeat

Trong trường hợp, khi hình ảnh nhỏ hơn kích thước hiển thị của phần tử mà ta thiết lập thì hình ảnh sẽ tự động được lặp lại để lắp đầy kích thước hiển thị của phần tử.

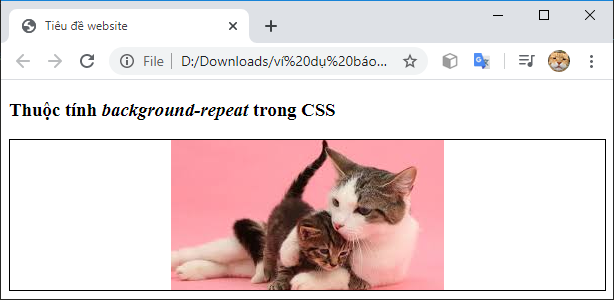


Hình : Hình ảnh sẽ bị lặp lại nếu kích thước ảnh nhỏ hơn kích thước hiển thị.

Để vô hiệu hóa sự lặp lại đó, ta sử dụng thuộc tính *background-repeat*.

<div style="background-image: url('images.png'); height:200px; background-position:right; background-repeat: no-repeat; "> Phần tử dùng ảnh để làm hình nền </div>

Kết quả khi hiển thị trên trình duyệt:



Hình : Sử dụng thuộc tính background-repeat loại bỏ việc lặp lại ảnh.

### Các thuộc tính định vị trí bố cục trong CSS

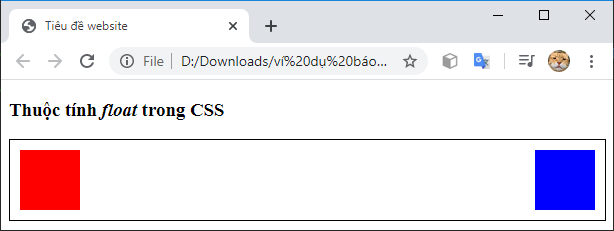
#### Float

*Float* (tiếng Việt nghĩa là thả nổi) đây là thuộc tính cố định một phần tử về bên trái hay bên phải không gian bao quanh nó. Mặc định giá trị của nó là *none*. Đây là thuộc tính quan trọng để dàn bố cục trang web, đặc biệt là dùng cho các phần tử *<div>*.

<div style="width: 60px; height: 60px; background-color: red; float: left;"></div>

<div style="width: 60px; height: 60px; background-color: blue; float: right;"></div>

Trong đoạn mã trên, ta có hai phần tử với hai màu đỏ và xanh lần lượt được thả nổi về bên trái và bên phải trong không gian bao quanh chúng.



Hình : Thả nổi phần tử với thuộc tính float để bố cục trang web.

#### Left, Right, Top, Bottom

Đây là các thuộc tính dùng để di chuyển đối tượng từ các hướng theo một giá trị được tính bằng pixel cho trước. Ví dụ sau sẽ di chuyển một đối tượng cách lề trái 50px, các thuộc tính còn lại cũng dùng tương tự.

<div style="width: 60px; height: 60px; background-color: red; left: 50px;"></div>

Kết quả khi hiển thị trên trình duyệt:



Hình : Di chuyển đối tượng với các thuộc tính left, right, top, bottom.

#### Position

*position* là một thuộc tính định vị đối tượng với hai giá trị: *absolute* là định vị tuyệt đối, nghĩa là vị trí của đối tượng sẽ được cố định một chỗ không thể di chuyển và giá trị thứ hai là *relative*, nghĩa là định vị tương đối, vị trị của đối tượng có thể thay đổi tùy theo độ rộng màn hình.

<div style="width: 60px; height: 60px; background-color: red; left: 50px; position: absolute;"></div>

#### Margin

Dùng để tạo khoảng cách giữa các đối tượng.Giá trị bên trong *margin* có thể là 4 giá trị xác định khoảng cách tương ứng theo thứ tự: *trên, phải, dưới, trái*. Ví dụ: *margin: 2px 5px 4px 7px*; Có nghĩa là từ viền đối tượng cách viền trang hoặc đối tượng khác về hướng trên là 2px, hướng phải là 5px, hướng dưới là 4px và hướng trái là 7px.

Giá trị bên trong *margin* cũng có thể có 2 giá trị. Khi đó khoảng cách sẽ được xác định tương ứng theo cặp: *trên và dưới, phải và trái*. Ví dụ: *margin: 2px 4px*; Có nghĩa là từ viền của đối tượng cách đối tượng khác hoặc viền trang theo hướng trên và dưới đều là 2px, theo hướng phải và trái đều là 4px.

Cũng có thể giá trị bên trong *margin* chỉ có 1 giá trị. Khi đó khoảng cách sẽ được xác định theo bộ tứ, tức là khoảng cách về 4 hướng của đối tượng so với đối tượng khác hoặc viền trang đều như nhau. Ví dụ: *margin: 4px*; Có nghĩa là từ viền của đối tượng đến viền trang hoặc đối tượng khác theo hướng trên là 4px, theo hướng phải cũng là 4px, theo hướng dưới cũng là 4px và theo hướng trái cũng là 4px.

#### Padding

Định khoảng cách từ nội dung đặt trong đối tượng đến viền của đối tượng đó (hay còn gọi là vùng đệm). Các giá trị thuộc tính cũng tương tự như margin.

## Ngôn ngữ Javascript

### Tổng quan về Javascript

JavaScript thường được viết tắt là JS, là một ngôn ngữ lập trình kịch bản phía máy khách (client-side) dựa vào đối tượng phát triển có sẵn hoặc tự định nghĩa, JavaScript được sử dụng rộng rãi trong các ứng dụng website. Cùng với [HTML](http://minhhn.com/wordpress/wordpress-va-html-lua-chon-nao-la-tot-nhat-cho-website-cua-ban/) và CSS, JavaScript là một trong ba công nghệ cốt lõi của World Wide Web. JavaScript được hỗ trợ hầu như trên tất cả các trình duyệt như Firefox, Chrome,… thậm chí với các trình duyệt trên thiết bị di động.

Khi duyệt trang Web, nếu trang Web cho phép người dùng nhập liệu, bạn cần viết một phương thức để kiểm soát dữ liệu nhập có hợp lệ hay không. Để thực hiện điều này, bạn có thể sử dụng một trong hai phương thức: kiểm tra tại trình duyệt của ngƣời dùng trước khi gửi dữ liệu về Server (dùng các Client Script như JavaScript, VBScript,…), gửi dữ liệu về máy chủ rồi mới kiểm tra và xử lý dữ liệu (dùng các Server Script như PHP chẳng hạn).

JavaScript là một ngôn ngữ thông dịch. Để sử dụng ngôn ngữ JavaScript, bạn cần phải tuân thủ một số quy tắc như:

* Trong JavaScript có sự phân biệt CHỮ HOA / chữ thường.
* Một câu lệnh đơn kết thúc bằng dấu chấm phẩy (;).
* Tập hợp nhiều câu lệnh đơn được đặt trong cặp dấu { } được gọi là một khối lệnh
* Ghi chú một dòng: //
* Ghi chú nhiều dòng: /\* \*/
* Dữ liệu là chuỗi, phải được đặt trong cặp dấu nháy kép ( “ “ ).
* Có thể dùng dấu + để nối các chuỗi hoặc các biến.
* Có thể đặt JavaScript ở bất cứ đâu trong tập tin HTML.
* Có thể tạo nhiều Script trong cùng một tập tin HTML.
* Có thể chèn thẻ HTML vào JavaScript (đặt trong nháy kép “ ” như một chuỗi).
* Có hai cú pháp để gọi JavaScript:

Nhúng trực tiếp vào trang HTML:

<script language=”JavaScript”>  
 //Các câu lệnh JavaScript;  
</script>

Gọi từ file bên ngoài:

<script language="javascript" src="URL "></script>

Trong đó, URL là đường dẫn đến file Javascipt. Tập tin này có phần mở rộng là \*.js.

### Một số ví dụ về Javascript có thể làm

Javascript có thể thay đổi nội dung HTML.

document.getElementById("demo").innerHTML = "Hello JS";

JavaScript chấp nhận cả dấu ngoặc kép và dấu ngoặc đơn.

document.getElementById('demo').innerHTML = 'Hello JS';

JavaScript có thể thay đổi giá trị thuộc tính HTML.

document.getElementById('myImage').src='pic\_bulbon.gif'

JavaScript có thể thay đổi kiểu HTML (CSS).

document.getElementById("demo").style.display = "none";

JavaScript có thể hiển thị các phần tử HTML.

document.getElementById("demo").style.display = "block";

### Biến, toán tử, toán hạng, hằng, hàm, lệnh trong Javascript

1.3.3.1 Biến

Biến trong JavaScript không cần xác định kiểu trƣớc, kiểu của dữ liệu sẽ được xác định tại thời điểm gán giá trị cho biến. Một số lưu ý khi đặt tên biến là tên biến phải bắt đầu bằng ký tự chữ cái hoặc ký tự \_, không dùng các ký tự đặc biệt để đặt tên biến:( , [ , { , # , & ,…, không bắt đầu bằng ký số, không dùng khoảng trắng, không được trùng với các từ khóa trong JavaScript.

Cách khai báo:

var tên\_biến;

//Hoặc

var tên\_biến = giá trị;

Biến trong JavaScript cũng phân làm 2 loại: Biến toàn cục và biến cục bộ.

Biến toàn cục được khai báo ngoài các hàm và có phạm vi hoạt động từ vị trí khai báo trở về sau trong chương trình.

Biến cục bộ được khai báo trong chương trình con hoặc bên trong hàm và chỉ có phạm vi hoạt động từ vị trí khai báo đến kết thúc chương trình con hoặc kết thúc hàm.

1.3.3.2 Toán tử

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=4; x\*5 | 20 |
| / | Chia | 5/2 | 2.5 |
| % | Lấy dƣ | 5%2 | 1 |
| ++ | Tăng 1 | x=5; x++ | 6 |
| -- | Giảm 1 | x=5; x-- | 4 |

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 |

Toán tử so sánh: Trả về True nếu đúng, trả về False nếu sai:

|  |  |  |  |
| --- | --- | --- | --- |
| TOÁN TỬ | CHỨC NĂNG | VÍ DỤ | KẾT QUẢ |
| = = | Bằng | 5==8 | return false |
| != | Không bằng | 5!=8 | return true |
| > | Lớn hơn | 5>8 | return false |
| < | Nhỏ hơn | 5<8 | return true |
| >= | Lớn hợn hoặc bằng | 5>=8 | return false |
| <= | Nhỏ hơn hoặc bằng | 5<=8 | return true |

Toán tử logic: Trả về True nếu đúng, trả về False nếu sai:

|  |  |  |  |
| --- | --- | --- | --- |
| TOÁN TỬ | CHỨC NĂNG | VÍ DỤ | KẾT QUẢ |
| && | Và | x = 6; y = 3 ;  (x < 10 && y > 1) | return true |
| || | Hoặc | x = 6 ; y = 3;  (x==5 || y==5) | return false |
| ! | Not | X = 6; y = 3;  !(x==y) | return true |

Toán tử điều kiện:

(Điều kiện) ? value1: value2;

Nếu điều kiện bên trong dấu () là đúng thì trả về value1. Ngược lại thì trả về value2.

1.3.3.3 Hằng số

Khác với các ngôn ngữ lập trình khác, JavaScript không có kiểu hằng số CONST. Giá trị của biến sẽ đƣợc xác định tại thời điểm gán biến cho đến khi biến đƣợc gán lại với giá trị mới.

1.3.3.4 Hàm

JavaScript hỗ trợ các hàm có sẵn hoặc bạn có thể tự định nghĩa một hàm mới. Các hàm có sẵn thông dụng:

Để xuất dữ liệu lên Web, bạn có thể sử dụng các hàm write() hoặc writeln().

Trong đó hàm writeln() sẽ kèm theo ký tự xuống hàng khi kết thúc chuỗi hiển thị.

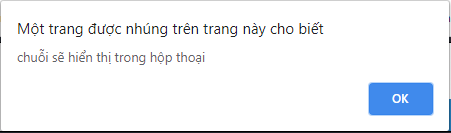
document.write(Nội dung cần hiển thị);

document.writeln(Nội dung cần hiển thị);

Hiển thị hộp thoại: có 3 dạng hộp thoại tùy theo dụng ý thông báo mà bạn có thể chọn một trong ba lọi này. Thứ nhất là hộp thoại OK, trên hộp thoại chỉ có duy nhất một nút OK kèm thông báo. Loại hộp thoại này thƣờng dùng để cảnh báo cho người dùng. Thứ hai là hộp loại YES/NO, trên hộp thoại có hai nút OK và Cancel. Loại hộp thoại này là dạng dành cho người dùng xác nhận thông tin, nếu đồng ý thì chọn OK, nếu không đồng ý thì chọn Cancel. Thứ ba là hộp thoại cho phép người dùng nhập giá trị vào. Trên hộp thoại dạng này có một khung nhập liệu, nút OK và Cancel. Đây là một dạng hộp thoại nhận thêm thông tin từ phía người dùng. Để xác nhận thông tin bổ sung, người dùng nhấn vào nút OK, nếu nhấn nút Cancel thì bỏ qua.

alert (“chuỗi sẽ hiển thị trong hộp thoại” );

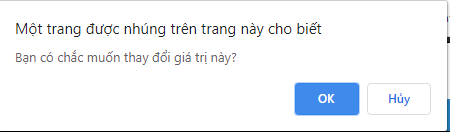
Kết quả hiển thị:



Hộp thoại OK

confirm (“Bạn có chắc muốn thay đổi giá trị này?”);

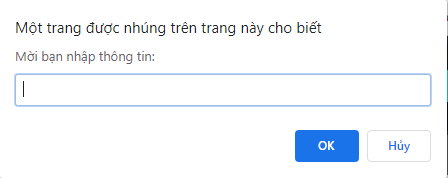
Kết quả hiển thị:



Hộp thoại YES/NO

prompt(“Mời bạn nhập thông tin:”);

Kết quả hiển thị:



Hộp thoại lấy thông tin

Các hàm xử lý chuỗi gồm hàm: để biến chuỗi (dạng biểu thức) thành biểu thức có thể tính toán đƣợc (hoặc trả về giá trị số của một chuỗi số) ta dùng hàm *eval(Chuỗi/biểu thức).* Trả về trị số nguyên của chuỗi số, nếu không có giá trị, hàm sẽ trả về NaN: *parseInt(“chuỗi\_số”)*. Ví dụ: nếu dùng *parseInt(“125.6a”);* thì kết quả trả về sẽ là 125. Nếu dùng *parseInt(“15ab6”);* thì kết quả trả về là 15 do 15 là hai ký số đầu tiên trong chuỗi, hàm sẽ hiểu có một phần số trong chuỗi và trả về số mà nó nhận diện đƣợc. Còn nếu dùng *parseInt(“dh154”);* thì kết quả trả về là NaN do chuỗi đƣa vào hàm parseInt lúc này có các ký tự đầu tiên không phải số, hàm sẽ dừng việc kiểm tra và thông báo đây không phải số. Tƣơng tự, chúng ta có hàm trả về trị số thực của chuỗi số, nếu không có giá trị, hàm sẽ trả về NaN: *parseFloat(“chuỗi\_số”)*. Nguyên tắc hoạt động của hàm này tƣơng tự nhƣ *parseInt* chỉ khác ở chỗ hàm *parseFloat* sẽ nhìn thấy dấu chấm động của số thực. Khi muốn kiểm tra nội dung ngƣời dùng nhập vào là chuỗi hay không, bạn dùng hàm *isNaN(giá\_trị\_kiểm\_tra).* Nếu *giá\_tri\_kiểm\_tra* không phải là số thì trả về True, nếu là số trả về False.

Ngoài ra, chúng ta có thể tự định nghĩa một hàm với cú pháp sau:

**function** tên\_hàm ( thamso1, thamso2,… )

{

//Khai báo các biến sử dụng trong hàm;

//Các câu lệnh xử lý trong JavaScript;

[return [giá trị /biểu thức] ];

}

Ví dụ hàm tự định nghĩa:

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<title>Ví dụ 3-1</title>

<script language="javascript">

function phepnhan(a,b)

{

x=a\*b;

return x;

}

</script>

</head>

<body>

<script language="javascript">

document.write(phepnhan(2, 6));

</script>

</body>

</html>

Lưu ý: Tên hàm tư định nghĩa không được trùng với từ khóa tên hàm trong JavaScript và không được trùng với tên các thành phần trong form HTML nếu đang truy vấn dữ liệu của form.

1.3.3.5 Các câu lệnh điều kiện trong JavaScript

Câu lệnh điều kiện if / if … else

Câu lệnh đơn giản, có 2 cú pháp:

if (<Biểu thức điều kiện>)

*{//Khối lệnh;}*

Cú pháp này dùng cho trường hợp chỉ cần kiểm tra điều kiện, nếu thoả điều kiện thị thực hiện công việc chỉ định trong khối lệnh. Nếu không thỏa điều kiện thì không làm gì cả.

Cú pháp 2:

if (<Biểu thức điều kiện>)

{//Khối lệnh 1;}

else

{//Khối lệnh 2;}

Cú pháp 2 áp dụng cho trường hợp nếu thỏa điều kiện thì thực hiện khối lệnh thứ nhất. Ngược lại nếu không thỏa điều kiện trong if sẽ thực hiện khối lệnh thứ 2.

Nếu có nhiều hơn 2 điều kiện, chúng ta có thể áp dụng các dòng if … else lồng nhau để có được cấu trúc điều kiện phức tạp

Cú pháp 3:

if(<biểu thức điều kiện1>)

{//Khối lệnh 1;}

else if (<biểu thức điều kiện 2>)

{//Khối lệnh 2;}

… else

{//Khối lệnh 3;}

Ví dụ: Một đoạn lệnh kiểm tra các dạng tam giác được viết bằng JavaScript

<script language="javascript">

a=eval(prompt("Nhập cạnh a"));

b=eval(prompt("Nhập cạnh b"));

c=eval(prompt("Nhập cạnh c"));

if(a + b < c || b + c < a || c + a < b)

alert("Không phải tam giác");

else if(a==b&&b==c&&c==a)

alert("Tam giác đều");

else if(a==b||b==c||c==a)

alert("Tam giác cân");

else alert("Tam giác thường");

</script>

Lựa chọn switch … case

Trường hợp có nhiều điều kiện, nếu phải dùng cấu trúc if … else lồng nhau sẽ làm cho code trở nên rườm rà, phức tạp. Cho nên, bạn có thể sử dụng cấu trúc lựa chọn switch … case cho các trường hợp như thế.

Cú pháp 1:

switch(Biểu thức)

{

case value1:

//Khối lệnh 1;

break;

case value2:

//Khối lệnh 2 ;

break;

………

case valuek:

//Khối lệnh k ;

break;

}

Với cú pháp 1, nếu biểu thức kiểm tra trong switch không thỏa mang bất kỳ giá trị nào trong các case, khi đó switch sẽ không trả kết quả.

Cú pháp 2:

switch(Biểu thức)

{

case value1:

//Khối lệnh 1;

break;

case value2:

//Khối lệnh 2 ;

break;

………

case valuek:

//Khối lệnh k ;

break;

default :

//Khối lệnh k+1;

break;

}

Nếu dùng cú pháp 2, khi giá trị của biểu thức không trùng với các giá trị liệt kê sẵn trong case thì switch sẽ thực hiện khối lệnh đặt trong default.

Ví dụ: dùng switch … case để giải bài toán xác định số ngày trong tháng

<script language="javascript">

t=prompt( "nhap thang: ");

switch (eval(t))

{

case 1:case 3:case 5:case 7:case 8:case 10:case 12:

alert("Thang "+ t + " co 31 ngay");

break;

case 2:

alert("Thang "+ t + " co 28 ngay hoặc 29 ngày");

break;

case 4: case 6: case 9: case 11:

alert("Thang "+t +" co 30 ngay");

break;

default:

alert("Khong co thang nay");

break;

}

</script>

Khi người dùng nhập vào một con số trong hộp thoại prompt. Nếu số nhập vào là một trong các số: 1, 3, 5, 7, 8, 10, 12 sẽ có cùng kết quả trả về là hộp thoại alert với thông báo tháng vừa nhập có 31 ngày. Nếu số mà người dùng nhập vào là số 2 thì sẽ xuất hộp thoại cho biết tháng 2 có 28 hoặc 29 ngày. Trƣờng hợp ngƣời dùng nhập một trong các số: 4, 6, 9, 11 thì kết quả sẽ là hộp thoại cho biết tháng vừa nhập có 30 ngày. Cuối cùng là nếu người dùng nhập các con số khác không thuộc các trường hợp trên thì đều xuất ra hộp thoại cho biết không có tháng này.

Vòng lặp for

Với giá trị được khởi tạo ban đầu của biến, kiểm tra biến thỏa điều kiện của vòng lặp thì thực hiện công việc. Sau khi kết thúc vòng lặp, giá trị của biến sẽ được thay đổi có thể tăng hay giảm tùy theo bài toán cụ thể. Sau khi được thay đổi giá trị, biến sẽ được kiểm tra lại nếu vẫn thỏa điều kiện thì vòng lặp tiếp tục được thực hiện. Nếu không còn thỏa điều kiện thì kết thúc vòng lặp.

Cú pháp:

for (biểu thức 1; biểu thức 2; biểu thức 3){

//Khối lệnh;

}

Trong đó, biểu thức 1 thƣờng là biểu thức khởi tạo biến, biểu thức 2 là biểu thức điều kiện, biểu thức 3 là biểu thức thay đổi giá trị của biến

Ví dụ: vận dụng dòng lặp for vẽ lên Web một bảng biểu có m dòng, n cột được nhập từ bàn phím.

<html>

<head>

<title> tạo bảng với m dòng, n cột nhập từ bàn phím </title>

</head>

<body>

<script language="javascript">

var n, m, i, j;

m=prompt("Nhap so dong");

n=prompt("Nhap so cot");

document.write("<table width=50% border=1>");

for(i=1;i<=m;i++)

{

document.write("<tr>"); for(j=1;j<=n;j++)

document.write("<td>" + i + j +"</td>");

document.write("</tr>");

}

document.write("</table>");

</script>

</body>

</html>

Trong ví dụ này, bạn có thể thấy cách chúng tôi gọi thẻ HTML nhƣ một chuỗi trong JavaScript. Khi đoạn Script được thông dịch, trình duyệt sẽ nhận diện các thẻ HTML trong chuỗi của JavaScript và hiển thị định dạng của thẻ HTML.

Vòng lặp for..in

Vòng lặp for..in thường được dùng cho trường hợp đối tượng là mảng. Cú pháp:

for ( variable in object ){

//khối lệnh

}

Ví dụ: Xuất dữ liệu trong mảng bằng for … in

<script language="javascript">

obj= new Array() ; // khai báo mảng

obj[0]="Xin ";

obj[1]="chào ";

obj[2]="mừng ";

obj[3]="các ";

obj[4]="bạn. ";

for(i in obj)

document.write(obj[i]);

</script>

Vòng lặp while

Vòng lặp while sẽ kiểm tra điều kiện nếu thỏa thì thực hiện khối lệnh trong while cho đến khi điều kiện không còn thỏa nữa. Chú ý rằng rất có thể dòng lặp while sẽ không được thực hiện lần nào nếu biểu thức không thỏa ngay từ đầu.

Cú pháp:

while(biểu thức điều kiện){

//Khối lệnh;

}

Ví dụ:

<script language="javascript">

var input;

while (input!=99)

{

input=prompt("Nhập vào một số bấy kỳ, nhập 99 đế thóat");

if (isNaN(input))

{

document.write("Dữ liệu không hợp lệ, nhập số ");

break;

}

}

</script>

Vòng lặp do … while

Dòng lặp do … while sẽ thực hiện công việc trƣớc rồi mới kiểm tra điều kiện. Có nghĩa là nếu điều kiện không thỏa thì công việc cũng đã được thực hiện ít nhất 1 lần.

Cú pháp:

do{

//khối lệnh;

}

while (biểu thức điều kiện);

Ví dụ:

<script language="javascript">

var input;

do

{

input=prompt(“Nhập vào một số bấy kỳ, nhập 99 để thóat”);

if(isNaN(input))

{

document.write(“Dữ liệu không hợp lệ, nhập số”);

break;

}

}while (input!=99)

</script>

# DANH MỤC TÀI LIỆU THAM KHẢO