**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

*Trà Vinh, ngày ….. tháng …… năm ……*

**Giáo viên hướng dẫn**

*(Ký tên và ghi rõ họ tên)*

**LỜI CẢM ƠN**

**NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN**

*Trà Vinh, ngày ….. tháng …… năm ……*

**Giáo viên hướng dẫn**

*(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, JAVASCRIPT

## Ngôn ngữ HTML

### Tổng quan HTML

HTML (HyperText Markup Language hay là "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 HTML 5.

### Lịch sử HTML

HTML được sáng tạo bởi [Tim Berners-Lee](https://en.wikipedia.org/wiki/Tim_Berners-Lee), nhà vật lý học của trung tâm nghiên cứu CERN ở Thụy Sĩ. Ông ta đã nghĩ ra được ý tưởng cho hệ thống hypertext trên nền Internet.

Hypertext có nghĩa là văn bản chứa links, nơi người xem có thể truy cập ngay lập tức. Ông xuất bản phiên bản đầu tiên của HTML trong năm 1991 bao gồm 18 thẻ HTML. Từ đó, mỗi phiên bản mới của HTML đều có thêm thẻ mới và attributes mới.

Theo Mozilla Developer Network: [HTML Element Reference](https://developer.mozilla.org/en-US/docs/Web/HTML/Element), hiện tại có hơn 140 thẻ HTML, mặc dù một vài trong số chúng đã bị tạm ngưng (không hỗ trợ bởi các trình duyệt hiện đại).

Nhanh chóng phổ biến ở mức độ chóng mặt, HTML được xem như là chuẩn mật của một website. Các thiết lập và cấu trúc HTML được vận hành và phát triển bởi World Wide Web Consortium (W3C).

Nâng cấp mới nhất gần đây là vào năm 2014, khi ra mắt chuẩn **HTML5.**Nó thêm vài thẻ vào, để xác định rõ nội dung như là <article>, <header> và <footer>.

### Phân lọai Website

#### Website tĩnh

Là những website không có phần quản trị nội dung. Nội dung trang web được chỉ định trong quá trình thiết kế.

Thiết kế website tĩnh sử dụng HTML, CSS và JavaScript .

* **Ưu điểm:**
* Người thiết kế có thể thiết kế giao diện theo kiểu mới lạ.
* Tốc độ truy cập nhanh bởi nó chỉ là những file HTML.
* Chi phí đầu tư thấp.
* **Nhược điểm:**
* Khó quản lý nội dung.
* Khó nâng cấp bảo trì.
* Mỗi khi cần thay đổi nội dung phải vào file HTML, CSS hoặc JavaScript  để chỉnh sửa.

#### Website động

Là website có phần quản trị nội dung. Nội dung trang web được tạo mới, cập nhật hoặc xoá dựa vào phần quản trị này.

Thiết kế website động sử dụng **HTML**, **JavaScript**, **CSS**, ngôn ngữ lập trình web (**PHP, Java, DotNet**,…) và **cơ sở dữ liệu**

* **Ưu điểm:**
* Dễ dàng quản lý nội dung.
* Dễ dàng nâng cấp và bảo trì.
* Có thể xây dụng được web lớn.
* Khả năng tương tác với người dùng cao.
* **Nhược điểm:**
* Chi phí xây dựng cao
* Nếu web lớn có thể cần thêm nhân sự chuyên ngành

### Cấu trúc trang HTML

<!DOCTYPE html>  
<html>  
<head>  
 <title>Page Title</title>  
</head>  
<body>  
 <h1>My First Heading</h1>  
 <p>My first paragraph.</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.

*<h1>* là thẻ dùng để khai báo độ lớn của tiều đề.

*<p>* là thẻ khai báo đoạn văn bản.

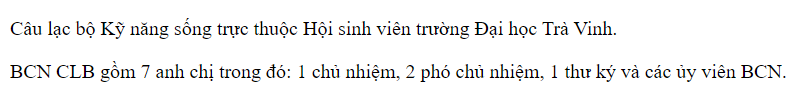
### Các thẻ HTML cơ bản

* **Thẻ <p> dùng để xác định một đoạn văn bản**

<p>Câu lạc bộ Kỹ năng sống trực thuocj Hội sinh viên trường Đại học Trà Vinh.</p>

<p>BCN CLB gồm 7 anh chị trong đó: 1 chủ nhiệm, 2 phó chủ nhiệm, 1 thư ký và các ủy viên BCN.</p>

Hiển thị:



* **Thẻ <h1> đến <h6> dùng để xác định tiêu đề**

<h1>Tiêu đề 1</h1>

<h2>Tiêu đề 2</h2>

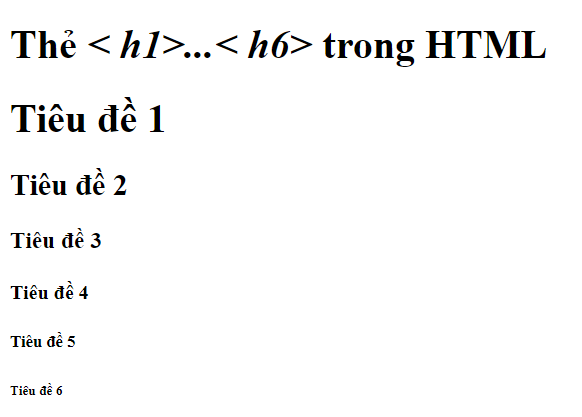
<h3>Tiêu đề 3</h3>

<h4>Tiêu đề 4</h4>

<h5>Tiêu đề 5</h5>

<h6>Tiêu đề 6</h6>

Hiển thị:



* **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="powderblue" width="50%" size="5">

Hiển thị:



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>

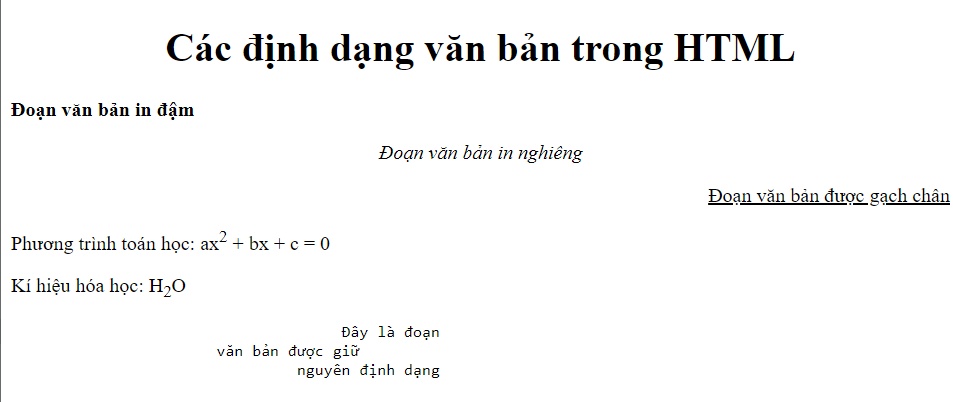
Đây là đoạn

văn bản được giữ

nguyên định dạng

</pre>

Hiển thị:



* **Thẻ <img> dùng để chèn hình vào trang web**

<img src="hinh/a.png" alt="Hình ảnh đơn vị" title="Thành viên CLB KNS">

Hiển thị:



* **Thẻ <a> dùng để tạo liên kết**

<a href="https://www.google.com.vn/">Truy cập vào google</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.

Hiển thị:



* **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>

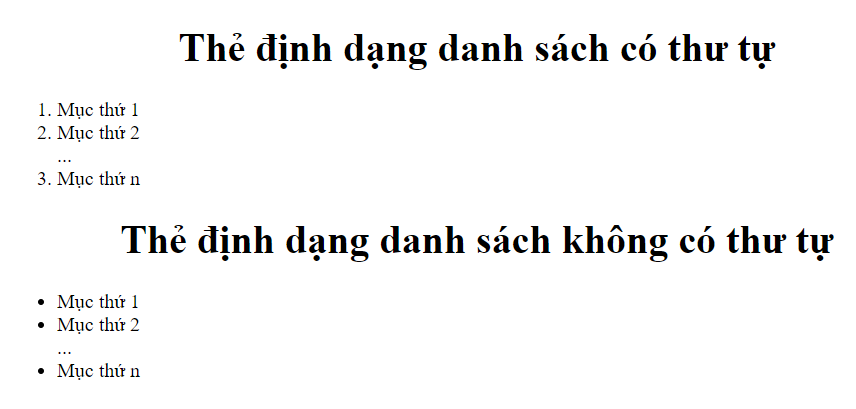
<li> Mục thứ 2 </li>

...

<li> Mục thứ n </li>

</ul>

Hiển thị:



* **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>

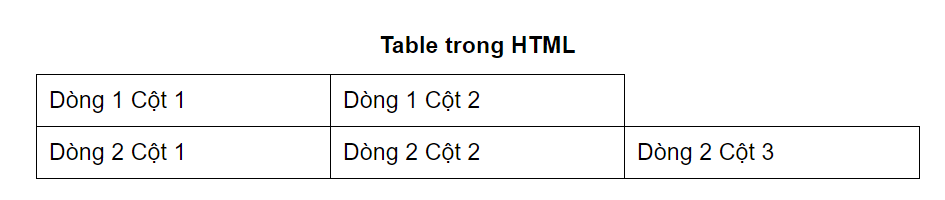
<td> Dòng 2 Cột 2 </td>

<td> Dòng 2 Cột 3 </td>

</tr>

</table>

Hiển thị:



* **Thẻ tạo form**

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.

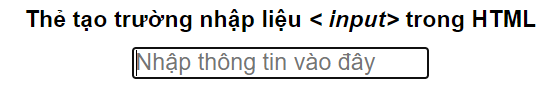
* **Thẻ input**

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

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.

Hiển thị:



Hình: Nhập dữ liệu với ô 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: blue"> Tiêu đề này sẽ có màu xanh dương </h1>

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.

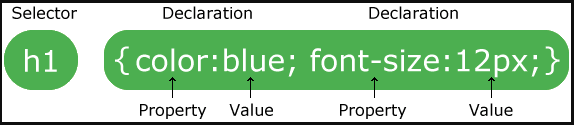
<head>

<link rel="stylesheet" href="myStyle.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: powderblue;

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

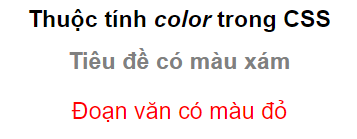
* **Color**

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.

<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 15: Thiết lập màu sắc với thuộc tính color CSS.

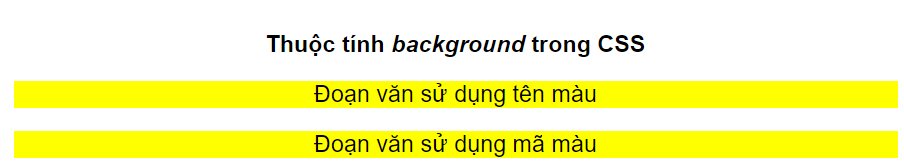
* **Background**

Thuộc tính background dùng để thiết lập màu nền cho một đoạn văn.

<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:



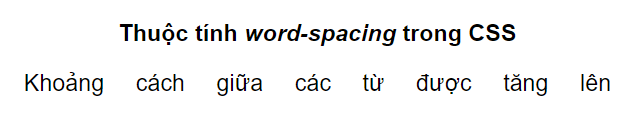
Hình 16: Thiết lập màu nền với thuộc tính background.

* **Word-spacing**

Thuộc tính word-spacing dùng để thiết lập khoảng cách giữa các từ.

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

Kết quả khi hiển thị trên trình duyệt:



Hình 17: Thiết lập khoảng cách các từ với thuộc tính word-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.

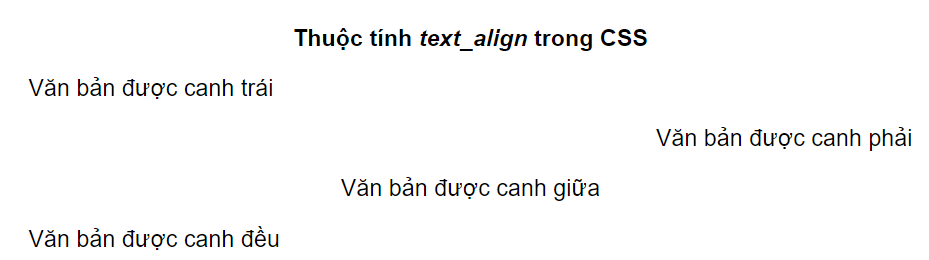
<p style="text-align: left"> Văn bản được canh trái</p>

<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:



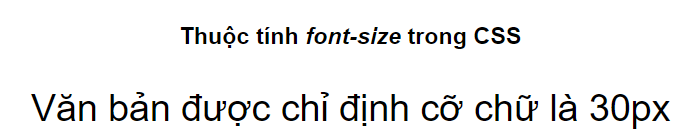
Hình 21: Chỉ định phông chữ với thuộc tính font-family.

* **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.

<p style="font-size: 30px;"> Văn bản được chỉ định cỡ chữ là 30px</p>

Kết quả hiển thị trên trình duyệt:

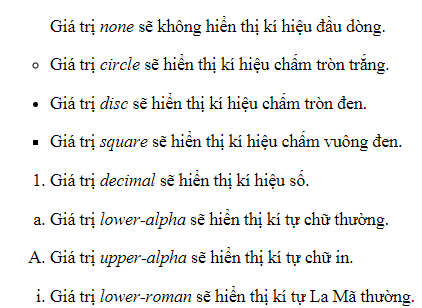


* **Thuộc tính định dạng danh sách trong CSS**

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,...) |
| **Thuộc tính** | **Giá trị** | **Ý nghĩa** |

Kết quả hiển thị trên trình duyệt:



Hình 23: Định kiểu danh sách với thuộc tính list-style.

* **Background-image**

Nếu muốn dùng hình ảnh để làm nền cho trang web, ta có thể dùng thuộc tính backround-image.

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

Kết quả hiển thị trên trình duyệt:

Hình 25: Thiết lập ảnh nền 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 hiển trị trên trình duyệt:



Hình 26: 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.



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; "></div>

Kết quả khi hiển thị trên trình duyệt:



Hình 28: Sử dụng thuộc tính background-repeat loại bỏ việc lặp lại ảnh.

## 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 trọng JavaScript

#### 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.

#### 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:

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

(Điều kiện) ? value1: value2;

#### 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.

#### Hàm

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

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

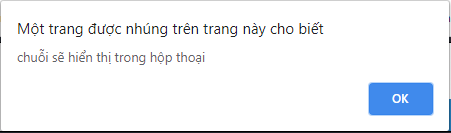
document.write(Nội dung cần hiển thị);

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” );

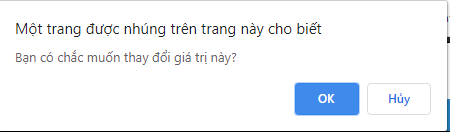
Kết quả hiển thị trên trình duyệt:



Hình: Hộp thoại OK

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

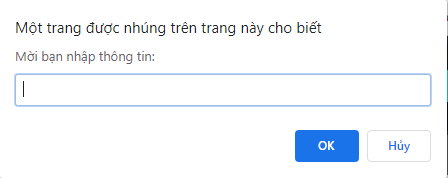
Kết quả hiển thị trên trình duyệt:



Hình: Hộp thoại YES/NO

prompt(“Mời bạn nhập thông tin:”);

Kết quả hiển thị trên trình duyệt:



Hình: 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 được (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 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. Tương 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 tương tự như *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.

#### Các câu lệnh điều kiện trong JavaScript

* Câu lệnh điều kiện if / if … else

Câu lệnh đơn giản, có 2 cú pháp:

Cú pháp 1:

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. Trường hợp người 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 thường 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 như 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

Vòng 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>

### Tham chiếu đến các đối tượng trong HTML

* Tham chiếu đến giá trị của thẻ

Để lấy được thông tin mà ngƣời dùng đã nhập vào form để kiểm tra tính hợp lệ, chúng ta cần tham chiếu đến các thành phần của form. Để tham chiếu đến form, bạn cần chú ý đến tên form và tên của các thành phần trong form mà bạn dự định sẽ lấy dữ liệu.

|  |  |
| --- | --- |
| **THẺ** | **Ví dụ** |
| Text | document.ten\_form.ten\_txt.value |
| Password | document.ten\_form.ten\_txtpass.value |
| Textarea | document.ten\_form.ten\_txtarea.value |
| Select (combobox) | document.ten\_form.ten\_combo.value |
| Select (listbox) | document.ten\_form.ten\_listbox.value |
| Checkbox | document.ten\_form.ten\_checkbox[i].value |
| Radio | document.ten\_form.ten\_radio[i].value |

* Tham chiếu đến chiều dài của thẻ

Để lấy được chiều dài của thông tin nhập nhiệu, chúng ta cần lấy được thông tin đó trước, rồi sau đó mới xem chiều dài của thông tin.

|  |  |
| --- | --- |
| **THẺ** | **VD** |
| Text | document.ten\_form.ten\_txt.value.length |
| **THẺ** | **VD** |
| Password | document.ten\_form.ten\_txtpass.value. length |
| Textarea | document.ten\_form.ten\_txtarea.value. length |
| Select (combobox) | document.ten\_form.ten\_combo.value. length |
| Select (listbox) | document.ten\_form.ten\_listbox.value. length |
| Checkbox | document.ten\_form.ten\_checkbox[i].value. length |
| Radio | document.ten\_form.ten\_radio[i].value. length |

* Tham chiếu đến trạng thái của thẻ

Chỉ có một số đối tượng trong form có thể kiểm tra trạng thái.

|  |  |
| --- | --- |
| **THẺ** | **VD** |
| Select (combobox) | document.ten\_form.ten\_combo.selected |
| Select (listbox) | document.ten\_form.ten\_listbox. selected |
| Checkbox | document.ten\_form.ten\_checkbox[i]. checked |
| Radio | document.ten\_form.ten\_radio[i]. checked |

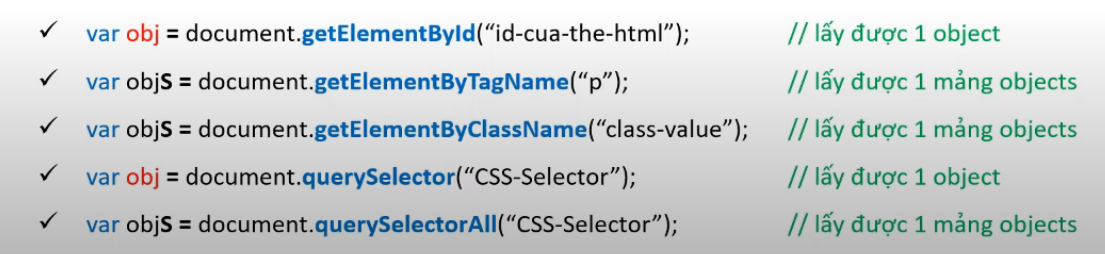
Nếu muốn tham chiếu đến trạng thái của một khung nhập liệu (chẳng hạn như Text Fiels, Password Field, …) có được nhập thông tin vào chưa, ta quay về với dạng tham chiếu đến giá trị của các Fields đó. Nếu sau khi tham chiếu mà trả về là rỗng tức là Field đó chưa được nhập liệu.

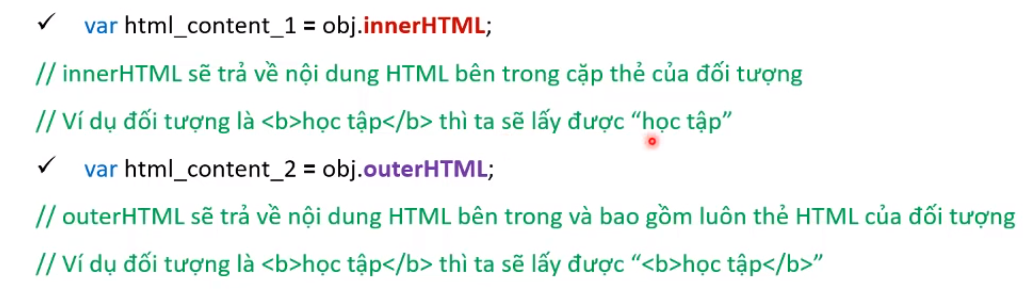
|  |  |
| --- | --- |
| **PHUƠNG THỨC** | **DIỄN GIẢI** |
| document.ten\_form.ten\_tp\_form.focus() | Đặt con trỏ nhấp nháy vào một thành phần có tên là ten\_tp\_form trong form có tên là ten\_form |
| **PHUƠNG THỨC** | **DIỄN GIẢI** |
| window.close() | Đóng của sổ hiện tại |
| window.open(URL) | Mở cửa sổ trình duyệt với địa chỉ URL |
| window.history.back(i) | Trở về cửa sổ trình duyệt trang web i lần |
| window.history.go(i) | Trở về cửa sổ trình duyệt trang web i lần |

### DOM trong JavaScript

Khi muốn viết nội dung vào các thành phần của form ta quay về dạng tham chiếu đến giá trị của thẻ và chỉ việc gán nội dung. Nhưng nếu đó là một thẻ HTML không thuộc thành phần của biểu mẫu, chúng ta không thể áp dụng theo cách tham chiếu giá trị được. Lúc này chúng ta sẽ sử dụng thuộc tính id để nhận dạng các thẻ HTML cần chèn thêm thông tin từ JavaScript. Sau đó sử dụng thành phần *getElementById(“ten\_id”)*. Tuy nhiên có hai dạng truy cập vào id của một thẻ HTML.

Một số phương thức về DOM:





# CHƯƠNG 2: TÌM HIỂU VỀ BOOTSTRAP

## Giới thiệu về Bootstrap

Bootstrap là một khung công tác miễn phí để phát triển web nhanh hơn và dễ dàng hơn.

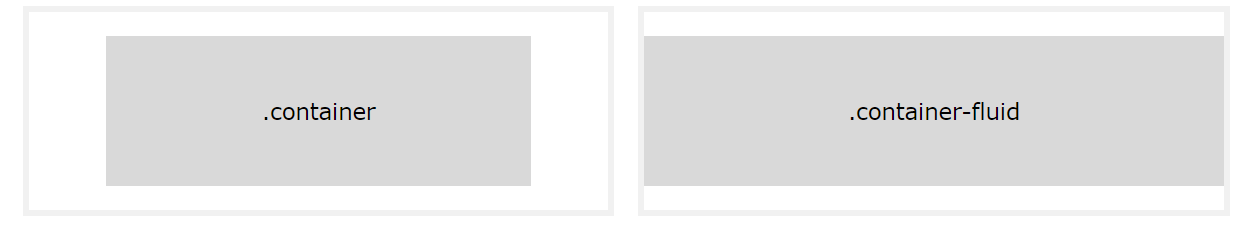
Bootstrap bao gồm các mẫu thiết kế dựa trên HTML và CSS cho kiểu chữ, biểu mẫu, nút, bảng, điều hướng, phương thức, băng chuyền hình ảnh và nhiều mẫu khác, cũng như các plugin JavaScript tùy chọn.

Bootstrap cũng cung cấp cho bạn khả năng dễ dàng tạo ra các thiết kế đáp ứng.

## Bootstrap container

Bootstrap yêu cầu một yếu tố có chứa để bọc nội dung trang web. Các **container** được sử dụng để đệm nội dung bên trong chúng và có hai lớp chứa:

* Các lớp **.container** đáp ứng **container** **có chiều rộng cố định.**
* Các lớp **.container-fluid** cung cấp **container** kéo dài toàn bộ chiều rộng của màn nhìn.

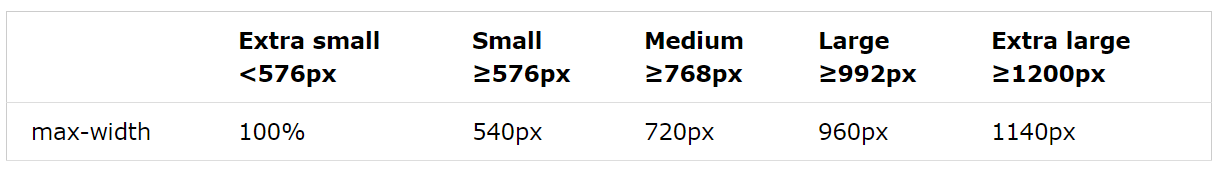


Hình : *Container, Container-fluid*

### Container cố định

Sử dụng lớp **.container** để tạo **container** có độ rộng cố định.

Lưu ý rằng chiều rộng của nó ( **max-width**) sẽ thay đổi trên các kích thước màn hình khác nhau.



Hình: Kích thước container

Ví dụ:

<div class="container">

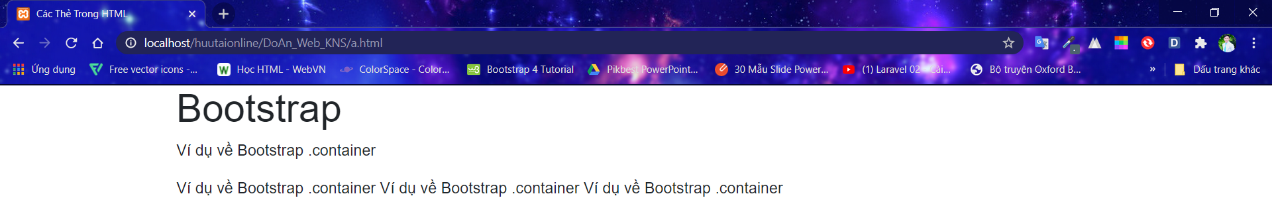
<h1>Bootstrap</h1>

<p>Ví dụ về Bootstrap .container</p>

<p>Ví dụ về Bootstrap .container</p>

</div>

Kết quả hiển thị:



*Hình :Sử dụng Container*

### Fluid Container

Sử dụng lớp **.container-fluid** để tạo **container** có toàn bộ chiều rộng của màn hình ( **width** luôn luôn **100%**)

Ví dụ:

<div class="container-fluid">

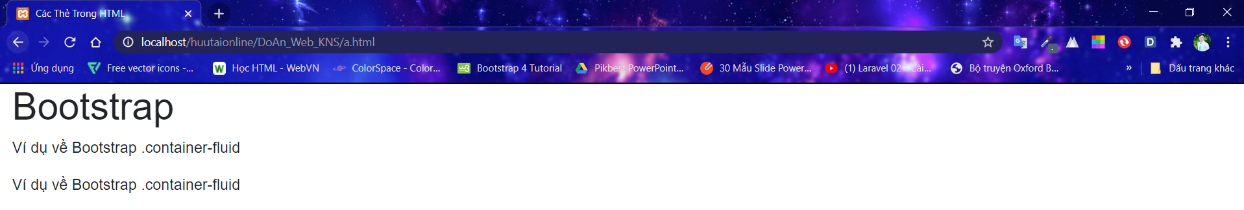
<h1>Bootstrap</h1>

<p>Ví dụ về Bootstrap .container-fluid </p>

<p>Ví dụ về Bootstrap .container-fluid </p>

</div>

Kết quả hiển thị:



Hình: Sử dụng Container-fluid

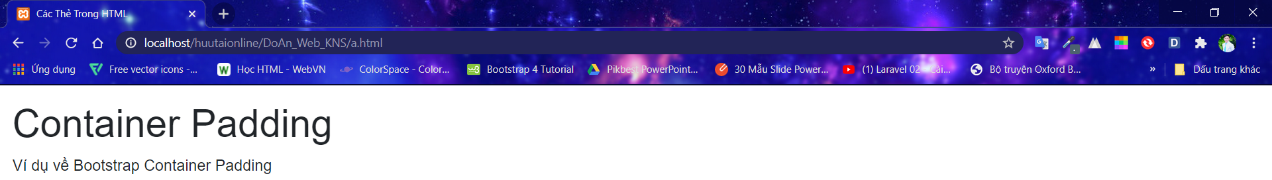
### Container Padding

Theo mặc định, các **container** có phần **padding** trái và phải 15px, không có phần padding trên cùng hoặc dưới cùng. Do đó, chúng tôi thường sử dụng **các tiện ích khoảng cách** , chẳng hạn như phần padding và margins để làm cho chúng trông đẹp hơn.

Ví dụ: **.pt-3** có nghĩa là "thêm phần padding trên cùng là 16px":

<div class="container pt-3"></div>

Kết quả hiển thị:



### Container Border và Color

Các tiện ích khác, chẳng hạn như đường viền và màu sắc, cũng thường được sử dụng cùng với các container:

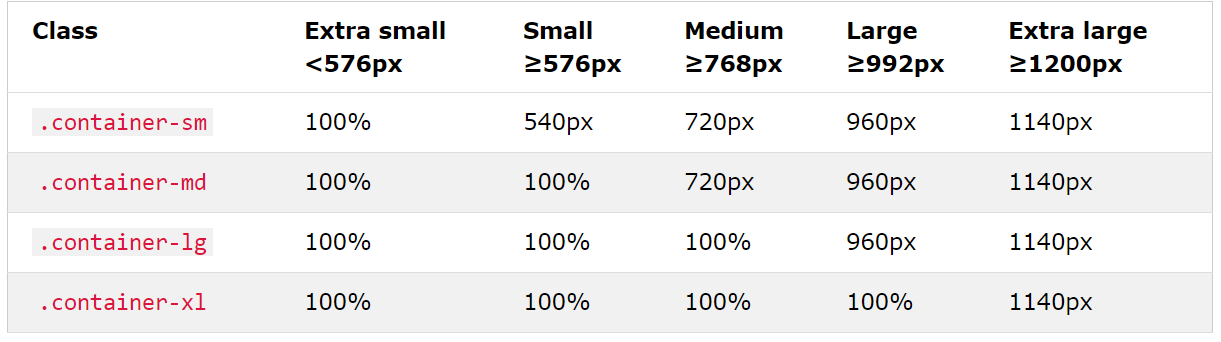
Ví dụ:

<div class="container p-3 my-3 border"></div>  
<div class="container p-3 my-3 bg-dark text-white"></div>  
<div class="container p-3 my-3 bg-primary text-white"></div

### Responsive Container

Bạn cũng có thể sử dụng các lớp **.container-sm|md|lg|xl** để tạo các responsive container.

Các **max-width** container sẽ thay đổi về kích thước màn hình khác nhau / viewports:



Hình: kích thước responsive container

Ví dụ:

<div class="container-sm border">.container-sm</div>

<div class="container-md mt-3 border">.container-md</div>

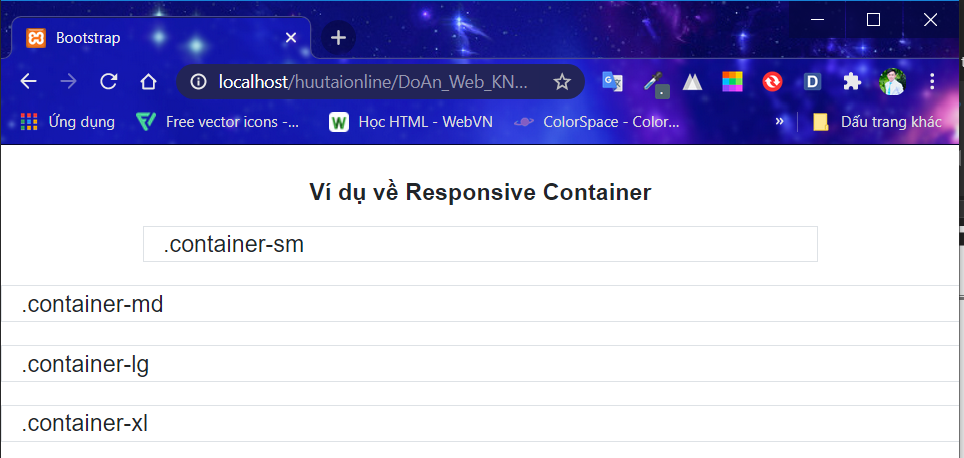
<div class="container-lg mt-3 border">.container-lg</div>

<div class="container-xl mt-3 border">.container-xl</div>

Kết quả hiển thị:



Hình:responsive container

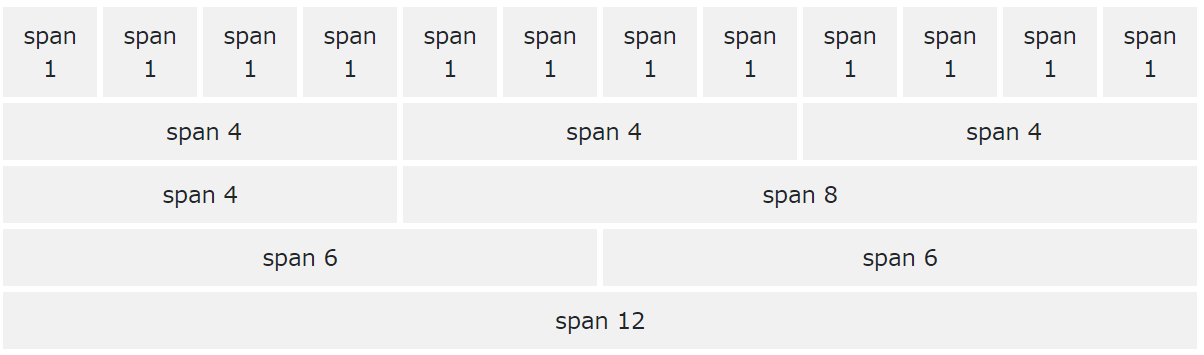


Hình: responsive container(tt)

## Bootstrap 4 Grids

Hệ thống Grids Bootstrap được xây dựng với flexbox và cho phép tối đa 12 cột trên trang.

Nếu bạn không muốn sử dụng tất cả 12 cột riêng lẻ, bạn có thể nhóm các cột lại với nhau để tạo các cột rộng hơn.



Hình 53: Hệ thống Grids

Hệ thống lưới phản ứng nhanh và các cột sẽ tự động sắp xếp lại tùy thuộc vào kích thước màn hình.

Đảm bảo rằng tổng cộng tối đa 12 hoặc ít hơn (không bắt buộc bạn phải sử dụng tất cả 12 cột có sẵn).

### Grid Classes

Hệ thống bootstrap 4 grids có năm lớp:

**.col-** (thiết bị nhỏ thêm - chiều rộng màn hình nhỏ hơn 576px)

**.col-sm-** (thiết bị nhỏ - chiều rộng màn hình bằng hoặc lớn hơn 576px)

**.col-md-** (thiết bị trung bình - chiều rộng màn hình bằng hoặc lớn hơn 768px)

**.col-lg-** (thiết bị lớn - chiều rộng màn hình bằng hoặc lớn hơn 992px)

**.col-xl-** (thiết bị xlarge - chiều rộng màn hình bằng hoặc lớn hơn 1200px)

Các lớp ở trên có thể được kết hợp để tạo ra bố cục năng động và linh hoạt hơn.

**Mẹo:** Mỗi lớp tăng tỷ lệ, vì vậy nếu bạn muốn đặt cùng độ rộng cho **sm** và **md**, bạn chỉ cần chỉ định **sm**.

### Cấu trúc cơ bản của Bootstrap 4 Grids

Sau đây là cấu trúc cơ bản của lưới Bootstrap 4:

<div class="row">

<div class="col-sm-6" style="background-color:yellow;">50%</div>

<div class="col-sm-6" style="background-color:orange;">50%</div>

</div>

<br>

<div class="row">

<div class="col-sm-4" style="background-color:yellow;">33.33%</div>

<div class="col-sm-4" style="background-color:orange;">33.33%</div>

<div class="col-sm-4" style="background-color:yellow;">33.33%</div>

</div>

<br>

<div class="row">

<div class="col-sm" style="background-color:yellow;">25%</div>

<div class="col-sm" style="background-color:orange;">25%</div>

<div class="col-sm" style="background-color:yellow;">25%</div>

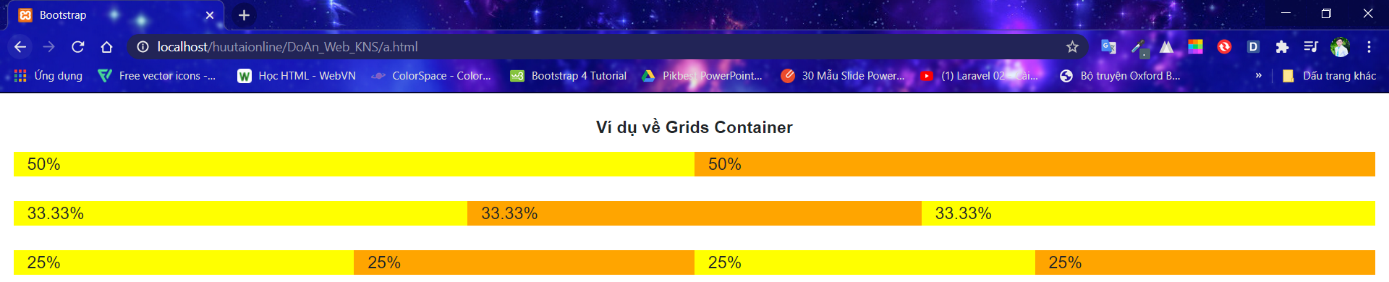
<div class="col-sm" style="background-color:orange;">25%</div>

</div>

Ví dụ đầu tiên: tạo một hàng **( <div class="row">** ). Sau đó, thêm số lượng cột mong muốn (thẻ với các **.col-\*-\***lớp thích hợp ). Ngôi sao đầu tiên (\*) đại diện cho khả năng phản hồi: **sm**, **md**, **lg** hoặc **xl**, trong khi ngôi sao thứ hai đại diện cho một số, nên thêm tối đa 12 cho mỗi hàng.

Ví dụ thứ hai: thay vì thêm một số cho mỗi số **col**, hãy để bootstrap xử lý bố cục, để tạo các cột có chiều rộng bằng nhau: hai "**col**"phần tử = 50% chiều rộng cho mỗi col. ba cols = 33,33% chiều rộng cho mỗi col. bốn cols = 25% chiều rộng, v.v. Bạn cũng có thể sử dụng **.col-sm|md|lg|xl** để làm cho các cột phản ứng nhanh.

Kết quả:



Hình 54: Cấu trúc cơ bản của Bootstrap 4 Grids

## Bootstrap 4 Alerts

Bootstrap 4 tạo các thông báo cảnh báo được xác định trước một cách dễ dàng:

Cảnh báo được tạo ra với các **.alert**, tiếp theo là một trong những lớp học theo ngữ cảnh  **.alert-success,  .alert-info,  .alert-warning,  .alert-danger,   .alert -primary, .alert-secondary, .alert-light** hoặc**.alert-dark**:

Ví dụ :

<div class="alert alert-success">

<strong>Success!</strong> Hộp cảnh báo này cho thấy một hành động thành công hoặc tích cực..

</div>

<div class="alert alert-info">

<strong>Info!</strong> Hộp cảnh báo này cho thấy một sự thay đổi hoặc hành động thông tin trung lập.

</div>

<div class="alert alert-warning">

<strong>Warning!</strong> Hộp cảnh báo này cho thấy một cảnh báo có thể cần chú ý.

</div>

<div class="alert alert-danger">

<strong>Danger!</strong>Hộp cảnh báo này cho thấy một hành động nguy hiểm hoặc có khả năng tiêu cực.</div>

<div class="alert alert-primary">

<strong>Primary!</strong> Hộp cảnh báo này cho thấy một hành động quan trọng.

.

</div>

<div class="alert alert-secondary">

<strong>Secondary!</strong> Hộp cảnh báo này cho thấy một hành động ít quan trọng hơn.

</div>

<div class="alert alert-dark">

<strong>Dark!</strong> Hộp cảnh báo màu xám đen..

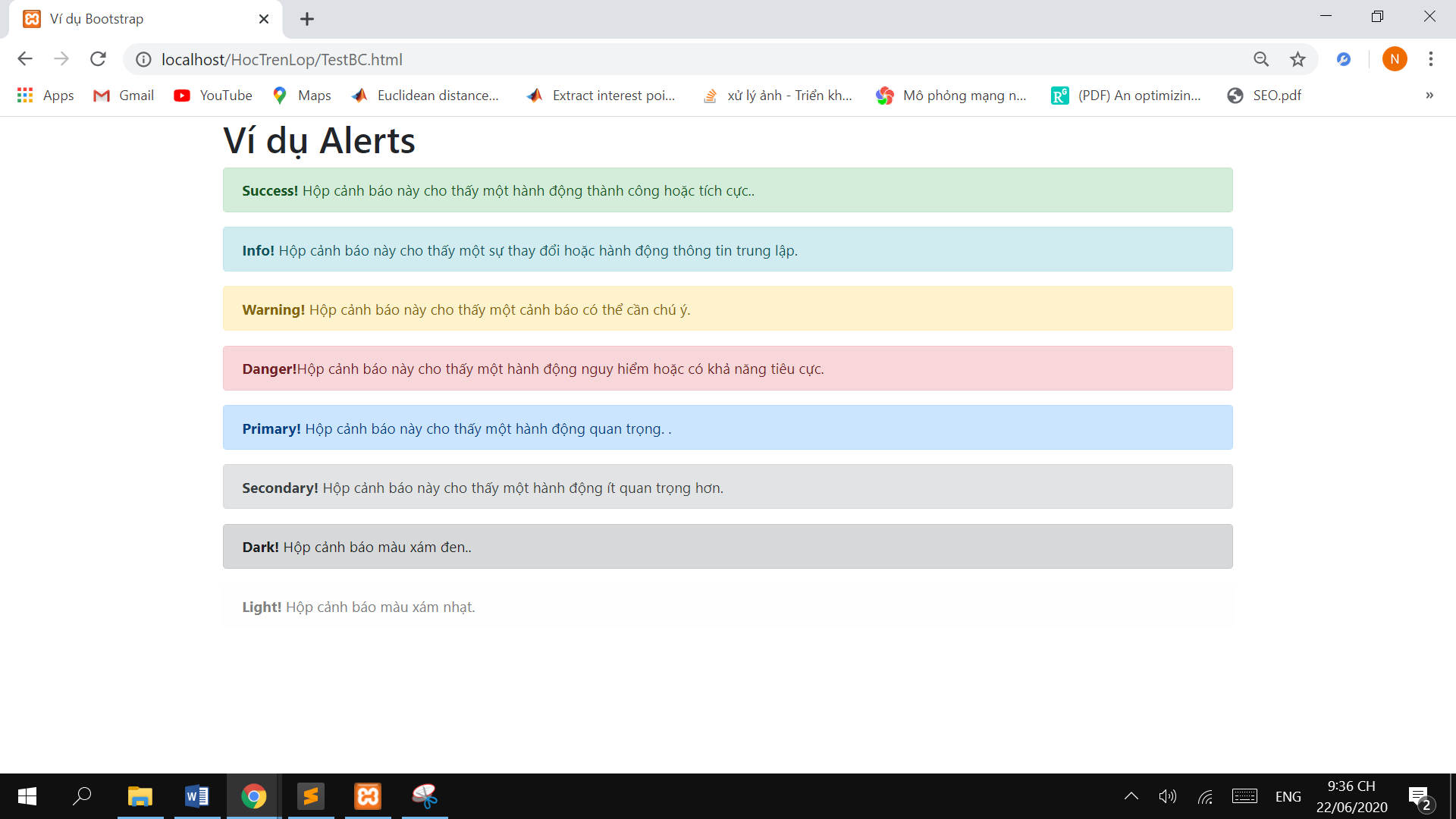
</div>

<div class="alert alert-light">

<strong>Light!</strong> Hộp cảnh báo màu xám nhạt.

</div></div>

Kết quả hiển thị:



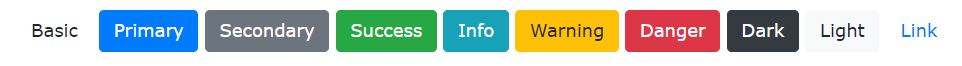
Hình 55: Bootstrap Alerts

## Bootstrap 4 Buttons

Bootstrap 4 cung cấp các kiểu button khác nhau như sau:

<button type="button" class="btn">Basic</button>  
<button type="button" class="btn btn-primary">Primary</button>  
<button type="button" class="btn btn-secondary">Secondary</button>  
<button type="button" class="btn btn-success">Success</button>  
<button type="button" class="btn btn-info">Info</button>  
<button type="button" class="btn btn-warning">Warning</button>  
<button type="button" class="btn btn-danger">Danger</button>  
<button type="button" class="btn btn-dark">Dark</button>  
<button type="button" class="btn btn-light">Light</button>  
<button type="button" class="btn btn-link">Link</button>

Kết quả hiển thị:



Hình 56: Bootstrap 4 Buttons

### Button Outline

<button type="button" class="btn btn-outline-primary">Primary</button>  
<button type="button" class="btn btn-outline-secondary">Secondary</button>  
<button type="button" class="btn btn-outline-success">Success</button>  
<button type="button" class="btn btn-outline-info">Info</button>  
<button type="button" class="btn btn-outline-warning">Warning</button>  
<button type="button" class="btn btn-outline-danger">Danger</button>  
<button type="button" class="btn btn-outline-dark">Dark</button>  
<button type="button" class="btn btn-outline-light text-dark">Light</button>

Kết quả hiển thị:



Hình 57: Button Outline

### Button Sizes

Sử dụng **.btn-lg** cho các nút lớn hoặc .btn-sm cho các nút nhỏ.

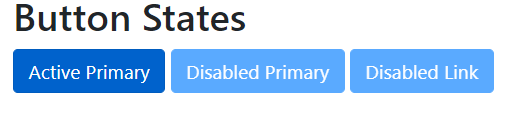
<button type="button" class="btn btn-primary btn-lg">Large</button>  
<button type="button" class="btn btn-primary">Default</button>  
<button type="button" class="btn btn-primary btn-sm">Small</button>

Kết quả hiển thị: 

Hình 58: Button Sizes

### Active/Disabled Buttons

Một nút có thể được đặt thành trạng thái hoạt động (xuất hiện nhấn) hoặc trạng thái bị vô hiệu hóa (không thể nhấn):



Hình 60: Active/Disabled Buttons

Lớp **.active** làm cho một nút xuất hiện khi được nhấn và thuộc tính **disabled** làm cho một nút không thể nhấn được. Lưu ý rằng phần tử **<a>** không hỗ trợ thuộc tính bị vô hiệu hóa và do đó phải sử dụng **.disabled** để làm cho nó xuất hiện trực quan bị vô hiệu hóa.

<button type="button" class="btn btn-primary active">Active Primary</button>  
<button type="button" class="btn btn-primary" disabled>Disabled Primary</button>  
<a href="#" class="btn btn-primary disabled">Disabled Link</a>

### Spinner Buttons

<button class="btn btn-primary">  
  <span class="spinner-border spinner-border-sm"></span>  
</button>  
  
<button class="btn btn-primary">  
  <span class="spinner-border spinner-border-sm"></span>  
  Loading..  
</button>  
  
<button class="btn btn-primary" disabled>  
  <span class="spinner-border spinner-border-sm"></span>  
  Loading..  
</button>  
  
<button class="btn btn-primary" disabled>  
  <span class="spinner-grow spinner-grow-sm"></span>  
  Loading..  
</button>

Kết quả hiển thị:



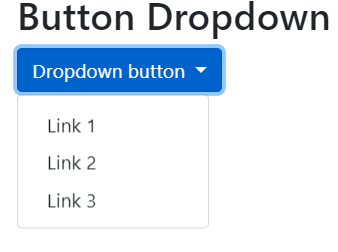
Hình 61: Spinner Buttons

## Bootstrap 4 Dropdowns

Menu thả xuống là một menu có thể bật cho phép người dùng chọn một giá trị từ danh sách được xác định trước:

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

Kết quả hiển thị:



Hình 62: Button Dropdown

Các lớp **.dropdown** cho thấy một trình đơn thả xuống.

Để mở **menu** **dropdown**, sử dụng nút hoặc liên kết với một lớp **.dropdown-toggle** và thuộc tính **data**-**toggle="dropdown**".

Thêm lớp **.dropdown-menu** vào một phần tử **<div>** để thực sự xây dựng menu thả xuống. Sau đó thêm lớp **.dropdown-item** vào từng thành phần (liên kết hoặc nút) trong menu dropdown.

### Dropdown Divider

Các lớp **.dropdown-divider** được sử dụng để liên kết riêng biệt bên trong menu dropdown có một đường kẻ ngang mỏng:

<div class="dropdown">

<button type="button" class="btn btn-primary dropdown-toggle" data-toggle="dropdown">

Lựa chọn

</button>

<div class="dropdown-menu">

<a class="dropdown-item" href="#">Thư</a>

<a class="dropdown-item" href="#">Hảo</a>

<a class="dropdown-item" href="#">Duy</a>

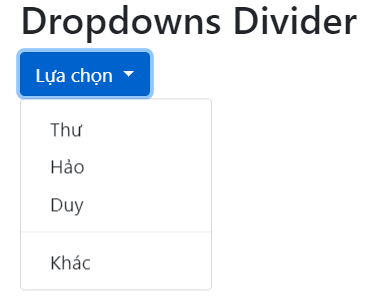
<div class="dropdown-divider"></div>

<a class="dropdown-item" href="#">Khác</a>

</div>

</div>

Kết quả hiển thị:



Hình 63: Dropdown Divider

### Dropdown Header

Các lớp **.dropdown-header** được sử dụng để thêm tiêu đề bên trong menu dropdown:

<div class="dropdown">

<button type="button" class="btn btn-primary dropdown-toggle" data-toggle="dropdown">

Dropdown Header button

</button>

<div class="dropdown-menu">

<h5 class="dropdown-header">Nhóm 1</h5>

<a class="dropdown-item" href="#">Thư</a>

<a class="dropdown-item" href="#">Hảo</a>

<a class="dropdown-item" href="#">Duy</a>

<h5 class="dropdown-header">Nhóm 2</h5>

<a class="dropdown-item" href="#">Nhi</a>

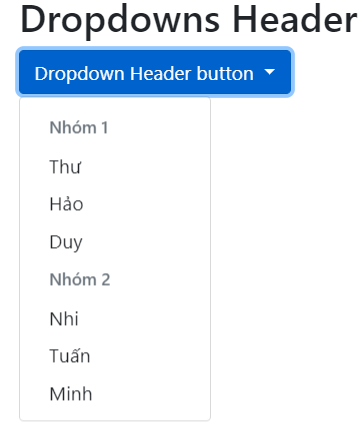
<a class="dropdown-item" href="#">Tuấn</a>

<a class="dropdown-item" href="#">Minh</a>

</div>

</div>

Kết quả hiển thị:



Hình 64: Dropdown Header

### Disable and Active items

Để vô hiệu hóa một mục trong menu thả xuống, hãy sử dụng **.disabled** (lấy màu văn bản màu xám nhạt và biểu tượng " **no-parking-sign** " khi di chuột):

<div class="dropdown">

<button type="button" class="btn btn-primary dropdown-toggle" data-toggle="dropdown">

Lựa chọn

</button>

<div class="dropdown-menu">

<a class="dropdown-item" href="#">Bình thường</a>

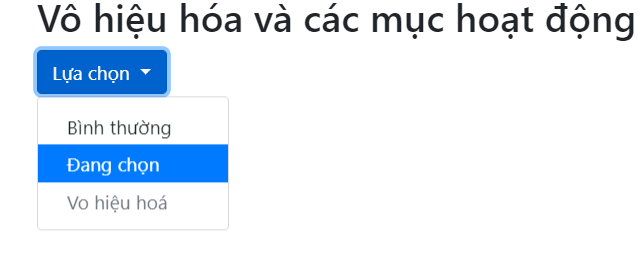
<a class="dropdown-item active" href="#">Đang chọn</a>

<a class="dropdown-item disabled" href="#">Vô hiệu hoá</a>

</div>

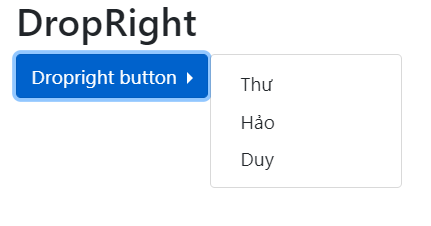
</div>

Kết quả hiển thị:



Hình 65: Disable and Active items

### Dropdown Position



Hình 66: Dropdown Position

## Bootstrap 4 Forms

### Cài đặt mặc định của Bootstrap 4

Tất cả các thuộc tính **<input>**, **<textarea>** và **<select>** các yếu tố với **.form-control** có chiều rộng 100%.

### Bootstrap 4 Bố cục mẫu

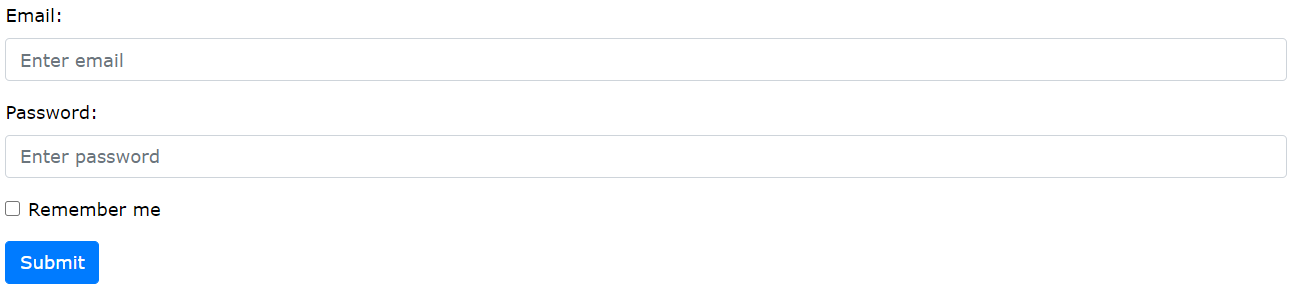
Bootstrap cung cấp hai loại bố cục biểu mẫu:

* Stacked (full-width) form
* Inline form

### Bootstrap 4 Stacked Form

<form action="/action\_page.php">  
  <div class="form-group">  
    <label for="email">Email address:</label>  
    <input type="email" class="form-control" placeholder="Enter email" id="email">  
  </div>  
  <div class="form-group">  
    <label for="pwd">Password:</label>  
    <input type="password" class="form-control" placeholder="Enter password" id="pwd">  
  </div>  
  <div class="form-group form-check">  
    <label class="form-check-label">  
      <input class="form-check-input" type="checkbox"> Remember me  
    </label>  
  </div>  
  <button type="submit" class="btn btn-primary">Submit</button>  
</form>

Kết quả hiển thị:



Hình 68: Bootstrap 4 Stacked Form

### Bootstrap Inline Form

Quy tắc bổ sung cho Inline form:

Thêm lớp **.form-inline** vào phần tử **<form>**

Ví dụ sau đây tạo một biểu mẫu nội tuyến với hai trường đầu vào, một hộp kiểm và một nút gửi:

<form class="form-inline" action="/action\_page.php">  
  <label for="email">Email address:</label>  
  <input type="email" class="form-control" placeholder="Enter email" id="email">  
  <label for="pwd">Password:</label>  
  <input type="password" class="form-control" placeholder="Enter password" id="pwd">  
  <div class="form-check">  
    <label class="form-check-label">  
      <input class="form-check-input" type="checkbox"> Remember me  
    </label>  
  </div>  
  <button type="submit" class="btn btn-primary">Submit</button>  
</form>

Kết quả hiển thị:



Hình 69: Inline Form

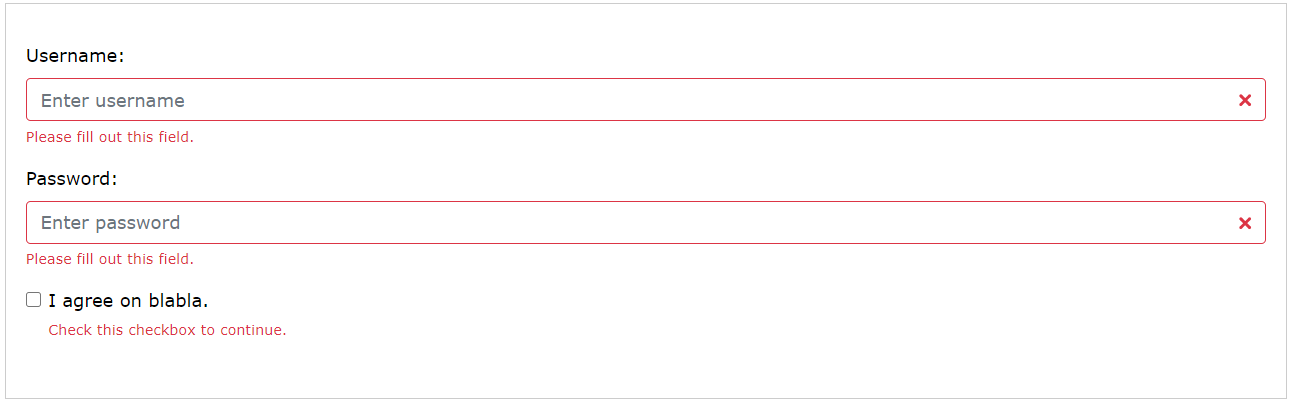
### Form Validation

Bạn có thể sử dụng các lớp xác nhận khác nhau để cung cấp phản hồi có giá trị cho người dùng. Thêm một **.was-validated** hoặc **.needs-validation** vào thành phần **<form>**, tùy thuộc vào việc bạn muốn cung cấp phản hồi xác thực trước hoặc sau khi gửi biểu mẫu. Các trường đầu vào sẽ có viền màu lục (hợp lệ) hoặc đỏ (không hợp lệ) để chỉ ra những gì còn thiếu trong biểu mẫu. Bạn cũng có thể thêm một **.valid-feedback** hoặc một **.invalid-feedback** thông báo để nói rõ cho người dùng biết những gì còn thiếu hoặc cần phải được thực hiện trước khi gửi biểu mẫu.

Trong ví dụ này, chúng tôi sử dụng **.was-validated** để chỉ ra những gì còn thiếu trước khi gửi biểu mẫu:

<form action="/action\_page.php" class="was-validated">  
  <div class="form-group">  
    <label for="uname">Username:</label>  
    <input type="text" class="form-control" id="uname" placeholder="Enter username" name="uname" required>  
    <div class="valid-feedback">Valid.</div>  
    <div class="invalid-feedback">Please fill out this field.</div>  
  </div>  
  <div class="form-group">  
    <label for="pwd">Password:</label>  
    <input type="password" class="form-control" id="pwd" placeholder="Enter password" name="pswd" required>  
    <div class="valid-feedback">Valid.</div>  
    <div class="invalid-feedback">Please fill out this field.</div>  
  </div>  
  <div class="form-group form-check">  
    <label class="form-check-label">  
      <input class="form-check-input" type="checkbox" name="remember" required> I agree on blabla.  
      <div class="valid-feedback">Valid.</div>  
      <div class="invalid-feedback">Check this checkbox to continue.</div>  
    </label>  
  </div>  
  <button type="submit" class="btn btn-primary">Submit</button>  
</form>

Kết quả hiển thị:



Hình 72: Form Validation

## Bootstrap 4 Icons

Bootstrap 4 không có thư viện biểu tượng riêng ( [Glyphicons từ Bootstrap 3](https://www.w3schools.com/bootstrap/bootstrap_glyphicons.asp) không được hỗ trợ trong BS4). Tuy nhiên, có nhiều thư viện biểu tượng miễn phí để lựa chọn, chẳng hạn như Biểu tượng thiết kế vật liệu tuyệt vời của Font Awesome và Google.

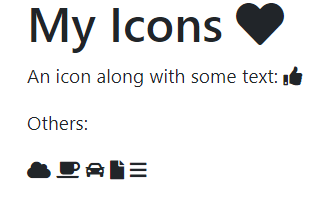
Để sử dụng các biểu tượng Font Awesome, hãy thêm phần sau vào trang HTML của bạn (Không cần tải xuống hoặc cài đặt)

<link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.0/css/all.css" integrity="sha384-lZN37f5QGtY3VHgisS14W3ExzMWZxybE1SJSEsQp9S+oqd12jhcu+A56Ebc1zFSJ" crossorigin="anonymous">

Sau đó, thêm tên của lớp biểu tượng vào bất kỳ thành phần HTML nội tuyến nào (như **<i>** hoặc **<span>**):

<i class="fas fa-cloud"></i>  
<i class="fas fa-coffee"></i>  
<i class="fas fa-car"></i>  
<i class="fas fa-file"></i>  
<i class="fas fa-bars"></i>

Kết quả hiển thị:



Hình 73: Bootstrap Icons

# CHƯƠNG 3: TỔNG QUAN VỀ PHP

## PHP là gì?

PHP có tên gọi đầy đủ là **P**ersonal **H**ome **P**age hay **H**ypertext **P**reprocessor là ngôn ngữ kịch bản trên Server (Server Side Script), được thực thi trên WebServer và kết quả được gửi trả về trình duyệt Web theo yêu cầu của người dùng.

PHP là một ngôn ngữ lập trình web động mã nguồn mở. Tập tin PHP có phần mở rộng là .php. Về tổng quan PHP có cú pháp khá tương đồng với một số ngôn ngữ như C, java. Tuy nhiên, tự bản thân chúng cũng có những điểm rất riêng biệt.

## Lịch sử phát triển PHP

PHP ra đời vào khoảng năm 1994 do Rasmus Lerdorf tạo ra với tên gọi là PHP/FI (Personal Home Page/Forms Interpreter) để theo dõi người dùng truy cập lý lịch trực tuyến của ông, đƣợc phát triển để phát sinh các form đăng nhập sử dụng giao thức HTTP của Unix.

PHP/FI 2 được công bố vào tháng 11 năm 1997 chuyển sang ngôn ngữ Script xử lý trên Server hỗ trợ cơ sở dữ liệu, Upload File, khai báo biến, mảng, câu lệnh điều kiện, biểu thức…

Năm 2005 (PHP 5): Bổ sung Zend Engine II hỗ trợ lập trình hƣớng đối tƣợng, XML, SOAP cho Web Services, SQLite.

Hiện nay phiên bản tiếp theo của PHP đang đƣợc phát triển và hi vọng sẽ lấp đầy những khuyết điểm của phiên bản hiện tại. Ví dụ: Hỗ trợ namespace, hỗ trợ Unicode…

## Ưu điểm và nhược điểm của PHP

### Ưu điểm

Là một ngôn ngữ mã nguồn mở, PHP được một số lớn các nhà lập trình Web chọn làm ngôn ngữ để phát triển Website nhờ vào một số ưu điểm nổi bật sau:

Đầu tiên đó là PHP được sử dụng miễn phí. Là một lợi thế cực lớn cho ai muốn học lập trình này. Các bạn có thể học và thực hành theo dạng online. Không cần phải lo đến việc chi trả số tiền lớn để học lập trình.

Cú pháp và cấu trúc của **PHP** tương đối dễ dàng. Nếu bạn muốn học về ngôn ngữ này, bạn sẽ không phải mất quá nhiều thời gian để hiểu được. Đa số các bạn lập trình viên thường ngại với cấu trúc khó. Là một ưu điểm lớn cho mọi người quan tâm và yêu thích hàng đầu về ngôn ngữ lập trình.

Sẽ rất dễ kiếm được việc làm với mức lương khủng. Đa số công ty thiết kế web hiện nay đều có bộ phận PHP. Và đó là điều không thể thiếu. Ngoài ra còn có thêm các công ty về phần mềm. Nhiều công việc lớn khác nhau, cũng như nhiều công ty khác nhau sẽ mang lại cho bạn thu nhập đáng kể.

Không chỉ với hiện tại mà trong tương lai. ngôn ngữ lập trình này sẽ còn phát triển mạnh me hơn nữa để khẳng định vị trí của mình luôn là cao nhất trong các bảng khảo sát ngôn ngữ lập trình.

### Nhược điểm

Bên cạnh những ưu điểm đã trình bày ở trên, PHP cũng có nhược điểm không thể chối bỏ. Đó là PHP được tạo ra chỉ để phục vụ một mục đích duy nhất là phát triển ứng dụng web. Do đó, bạn không thể dùng ngôn ngữ này để phát triển phần mềm ứng dụng. Nhược điểm này làm cho PHP mất đi một số lượng khá lớn khách hàng khi họ chỉ muốn phát triển ứng dụng chạy trên window-form.

## Cơ chế hoạt động của Web động

Một trang web chỉ có các thẻ HTML, JavaScript chỉ là một trang web tĩnh. Chúng rất ít có sự tương tác với người dùng. Nếu có cũng chỉ là các bước xử lý ở mức Client, ngay tại chính trình duyệt của người dùng như kiểm tra dữ liệu hoặc chỉ hiển thị cục bộ những gì trình duyệt xử lý đƣợc từ thông tin mà người dùng nhập vào. Điều này rất hạn chế khi các thông tin của bạn chỉ có thể ở phía cục bộ, không thỏa mãn như cầu chia sẻ thông tin. Hơn thế nữa, những gì bạn muốn hiển thị trên web tĩnh sẽ được viết trực tiếp trong trang HTML. Chính vì thế thông tin hiển thị cũng dễ bị lỗi thời, khó cập nhật và tốn nhiều thời gian trong quá trình soạn thảo cũng như cập nhật mới.

Một hƣớng khác để xây website đó là có sự tương tác thật sự giữa người dùng và trang web. Người dùng có thể đăng ký làm thành viên của một Website, có thể đăng nhập hệ thống website, thêm, sửa, xóa một số thông tin mà mình được cấp quyền. Những người dùng khác có thể xem những thay đổi đó trên website (nếu đƣợc phép). Người ta gọi đây là các trang web động. Như vậy, đòi hỏi web động phải có một cơ sở dữ liệu đi kèm để lưu trữ những hoạt động trên hệ thống Website. Như vậy, điều dễ nhận thấy sự khác biệt giữa web tĩnh và web động đó chính là về mặt lưu trữ và hiển thị thông tin lưu trữ.

Bên cạnh đó, web tĩnh và web động còn khác nhau về công cụ xây dựng. Như đã trình bày, web tĩnh được viết bằng ngôn ngữ HTML, CSS, JavaScript, … Web động bên cạnh chấp nhận tất cả các ngôn ngữ xây dựng web tĩnh, nó còn có ngôn ngữ dùng để xử lý dữ liệu chạy trên máy chủ: PHP, JSP, ASP,…

## Giới thiệu và cài đặt WebServer

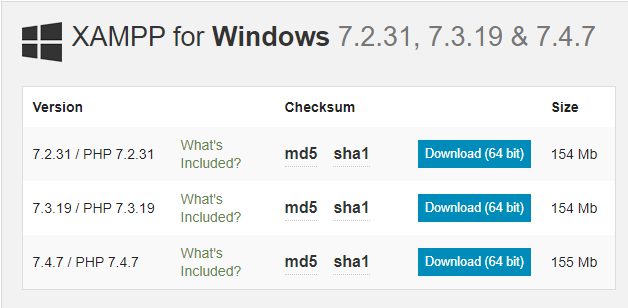
### Giới thiệu về WebSever

WebServer hay còn gọi là trình phục vụ Web là một máy tính đóng vai trò làm máy chủ có dung lƣợng lớn, tốc độ xử lý cao và có khả năng lưu trữ thông tin nhƣ một ngân hàng dữ liệu, chứa những Website đã đƣợc thiết kế cùng với những thông tin liên quan khác: các mã kịch bản, các chương trình, và các tập tin đa phương tiện, … Khi nhận được yêu cầu tìm một trình duyệt Web trên máy ngƣời dùng, WebServer sẽ xử lý yêu cầu đó và có khả năng gửi đến máy khách những trang Web thông qua môi trƣờng Internet (hoặc Intranet) qua giao thức HTTP - giao thức đượcc thiết kế để gửi các tập tin đến trình duyệt Web (Web Browser), và các giao thức khác.

Tất cả các WebServer đều có một địa chỉ IP hay một tên miền. Giả sử khi bạn gõ vào thanh địa chỉ của trình duyệt: [http://www.tvu.edu.vn](http://www.tvu.edu.vn/) và nhấn Enter. Yêu cầu của bạn sẽ được chuyển đến Server đang chứa Website của Trường Đại học Trà Vinh dựa vào tên miền mà bạn đã gõ. Server này sẽ tìm trang Web có tên là index, nếu bên trong tập tin này có kèm theo các mã HMTL và các Client Script ví dụ như JavaScript, thì Server sẽ gửi nó đến trình duyệt của bạn. Trình duyệt của bạn sẽ làm nhiệm vụ dịch và hiển thị nội dung, định dạng của trang theo ý định của ngƣời lập trình. Nếu trong tập tin bạn yêu cầu có các Server Script như PHP, JSP, ASP, … thì trình phục vụ Web sẽ dịch và gửi kết quả về cho trình duyệt của bạn hiển thị.

Bất kỳ một máy tính nào cũng có thể trở thành một WebServer bởi việc cài đặt lên nó một chương trình phần mềm WebServer Software: IIS, Apache, Netscape, ...và sau đó kết nối vào Internet là không quá khó. Tuy nhiên không phải máy tính nào cũng đƣợc chọn làm WebServer vì người ta còn quan tâm đến khả năng phục vụ, tốc độ, … Giống nhƣ những phần mềm khác mà bạn đã từng cài đặt trên máy tính của mình, WebServer Software cũng chỉ là một phần mềm ứng dụng. Nó được cài đặt, và chạy trên máy tính dùng làm WebServer, nhờ có chƣơng trình này mà ngƣời sử dụng có thể truy cập đến các trang Web từ một máy tính khác ở trên mạng. Ngày nay, WebServer Software còn có thể được tích hợp với cơ sở dữ liệu (Database), hay điều khiển việc kết nối vào cơ sở dữ liệu được chứa trên một Database Sever khác. Một số WebServer Software được tích hợp sẵn, có thể chạy trên máy tính cục bộ: WampServer, EasyPHP, AppServ, Xamp, …

Trong khuôn khổ tài liệu này, chúng tôi sẽ sử dụng bộ ba: PHP, Apache và MySQL để thực hiện xây dựng web động. Sau đây chúng tôi sẽ hướng dẫn các bạn cài đặt Xampp. Bạn có thể tải phiên bản mới nhất của Xampp tại địa chỉ: <https://www.apachefriends.org/download.html> và lựa chọn phiên bản tương ứng mà máy tính bạn đang sử dụng (ở đây sử dụng hệ điều hành Window).



Tệp tin tải về có tên là **xampp-windows-x64-7.4.7-0-VC15-installer.exe**

### Cài đặt Xampp

Để bắt đầu cài đặt Xampp bạn nhấp đúp vào tập tin **xampp-windows-x64-7.4.7-0-VC15-installer.exe**

### Sử dụng Xampp

## PHP cơ bản

### Quy ước trong PHP

Mã lệnh PHP được chèn vào vị trí nào trong trang HTML sẽ được biên dịch và hiển thị kết quả tại vị trí chèn mã lệnh. Có thể gọi mã lệnh PHP bằng nhiều cách. Một đoạn mã lệnh PHP có thể được đặt trong các cặp thẻ sau:

|  |  |
| --- | --- |
| **Thẻ mở** | **Thẻ đóng** |
| <? | ?> |
| <?php | ?> |
| <script language=”php”> | </script> |

Trong PHP phân biệt chữ hoa và chữ thường và tất cả các câu lệnh php đều cách nhau bởi dấu chấm phẩy (;). PHP không phân biệt khoảng trắng, Tab, xuống dòng.

Để ghép chuỗi trong PHP ta dùng dấu chấm (.)

Việc ghi chú lại ý nghĩa của dòng lệnh trong các ngôn ngữ lập trình là một vấn đề hết sức cần thiết. Ghi chú lại giúp chúng ta dễ dàng biết được chúng ta viết lệnh này dùng để làm gì phục vụ cho công việc gì. Trong PHP có 3 cách ghi chú:

// Đây là ghi chú một dòng

/\*

Đây là ghi chú nhiều dòng

\*/

Ví dụ:

|  |  |
| --- | --- |
| **Mã lệnh** | **Kết quả** |
| <?php // Khoi tao bien A $A=5; $a=10; /\*Hien thi bien A va bien a ra man hinh \*/ echo "A= ".$A." a= ".$a; ?> | A= 5 a= 10 |

### Khai báo biến

Biến trong PHP không cần khai báo kiểu dữ liệu, kiểu của biến sẽ được xác định tại thời điểm gán trị cho biến. Cú pháp khai báo như sau:

$ten\_bien=giá trị;

Trong đó, tên biến:

Có thể bao gồm các ký tự (A đến Z, a đến z), ký tự số (0…9), \_ , $

Không được khai báo biến bắt đầu bằng ký số (0…9)

Phân biệt chữ HOA – chữ thường

Ví dụ: khai báo biến a có giá trị là 5 ($a=5;).

|  |  |
| --- | --- |
| **Mã lệnh** | **Kết quả** |
| <?php  $ly=5;  $mon= “ly”;  echo “Diem mon” . $mon.  “ la “ . $$mon;  ?> | Diem mon ly la 5 |

### Kiểu dữ liệu

Biến trong PHP có thể lưu bất kỳ kiểu dữ liệu nào.

Các kiểu dữ liệu gồm: boolean (bool), integer (int), double (float, real), string, array, object.

### Kiểu chuỗi

Toán tử nối chuỗi: dấu chấm (.)

Ví dụ:

$s=”xin chao”.” ban”; // $s=”xin chao ban”;

Phân biện dấu nháy đơn và nháy kép: Các biến nằm trong chuỗi với dấu nháy kép sẽ được tính toán giá trị.

Ví dụ:

$t=”Tam”;

echo ‘ Chao $t‘; // Chao $t

echo “ Chao $t”; // Chao Tam

#### Một số hàm xử lý chuỗi

* Hàm strlen(): được dùng để trả về chiều dài của một chuỗi

Ví dụ:

|  |  |
| --- | --- |
| Mã lệnh | Kết quả |
| <?php  $s= “Hello world!!!”;  echo strlen($s);  ?> | 14 |

* Hàm strpos(): được sử dụng để tìm kiếm một ký tự trong chuỗi. Nếu hàm tìm thấy sẽ trả về vị trí của ký tự, ngược lại sẽ trả về false.

Ví dụ:

|  |  |
| --- | --- |
| Mã lệnh | Kết quả |
| <?php  $s= “Hello world!!!”;  echo strpos($s,  “world”);  ?> | 6 |

* Một số hàm xử lý ngày
* Lấy thời gian

Để tạo một nhãn thời gian, bạn sử sử dụng hàm **time** hoặc **mktime**. Hàm time tạo nhãn thời gian cho ngày tháng giờ hiện tại, còn hàm mktime tạo nhãn thời gian cho giá trị mà bạn truyền vào. Những hàm này đều trả về nhãn thời gian. Để định dạng lại nhãn thời gian, bạn có thể sử dụng hàm date như được hướng dẫn ở minh họa phía trên.

Khi sử dụng hàm mktime, các tham số không bắt buộc. Vì vậy, bạn có thể bỏ qua các tham số (tính từ tham số cuối) và chỉ làm việc với các tham số đầu. Nếu bỏ qua tham số, hàm mktime sử dụng giá trị hiện tại. Ví dụ, tham số cuối cùng định nghĩa năm, vì thế nếu bỏ qua tham số này, hàm mktime sẽ sử dụng năm hiện tại cho nhãn thời gian. Tương tự, nếu bỏ qua 2 tham số cuối, hàm mktime sử dụng năm và tháng hiện tại.

|  |  |
| --- | --- |
| Tên hàm | Mô tả |
| time() | Trả về ngày giờ hiện tại dƣới dạng nhãn thời gian |
| mktime{[$h[, $m[, $s[,  $M[, $D[, $Y]]]]]]} | Trả về nhãn thời gian dựa vào ngày tháng và thời gian truyền vào. Nếu bất cứ phần nào bị bỏ qua thì nó sẽ lấy giá trị của ngày giờ hiện tại. |
| checkdate($M, $D, $Y) | Trả về true nếu tháng, ngày, năm là hợp lệ. |
| getdate([$ts]) | Nếu bỏ qua tham số, hàm lấy ngày giờ hiện tại. |

* Định dạng thời gian

$date=date(“format”,$timestamp);

Ý nghĩa: Trả về chuỗi mô tả ngày tháng đã được định dạng theo chuỗi định dạng

$format. Mặc định, hàm làm việc với ngày giờ hiện tại. Tuy nhiên, bạn có thể dùng biến thứ hai để định nghĩa một nhãn thời gian cho bất kỳ ngày giờ nào.

Ví dụ:

$today=date(“Y/m/d”); $now=date(“h:i:s”);

* Mã định dạng thông dụng của hàm date

|  |  |  |  |
| --- | --- | --- | --- |
| Ký tự | Mô tả | Từ | Tới |
| D | Ngày trong tuần – 3 chữ cái | Mon | Sun |
| l | Ngày trong tuần – đủ chữ | Monday | Sunday |
| w | Ngày trong tuần (với 0=Sunday) | 0 | 6 |
| z | Ngày trong năm | 0 | 365 |
| S | Số thứ tự của ngày trong tháng (st, nd, rd, th) |  |  |
| n | Tháng – không có số 0 ở đầu | 1 | 12 |
| m | Tháng – có 0 ở đầu | 01 | 12 |
| M | Tháng – 3 chữ cái đầu | Jan | Dec |
| F | Tháng – đủ chữ | January | December |
| j | Ngày trong tháng – không có số 0 ở đầu | 1 | 31 |
| d | Ngày trong tháng – có 0 ở đầu | 01 | 31 |
| y | Năm – 2 chữ số | 10 |  |
| Y | Năm – 4 chữ số | 2010 |  |
| L | Năm nhuận (1) hoặc năm thƣờng (0) | 0 | 1 |
| g | Giờ - định dạng 12-giờ, không có 0 đầu | 1 | 12 |
| G | Giờ - định dạng 24-giờ, không có 0 ở đầu | 0 | 23 |
| h | Giờ - định dạng 12-giờ, có 0 ở đầu | 01 | 12 |
| H | Giờ - định dạng 24-giờ, có 0 ở đầu | 00 | 23 |
| i | Phút – có 0 | 00 | 59 |
| s | Giây – có 0 | 00 | 59 |
| a | am/pm – viết thƣờng | am | pm |
| A | AM/PM – viết hoa | AM | PM |
| T | Viết tắt cho vùng thời gian | EST |  |
| U | số giấy kể từ kỷ nguyên Unix | -2,147,483,648 | 2,147,483,647 |

### Các phép toán trong PHP

#### Các phép toán số học

|  |  |  |  |
| --- | --- | --- | --- |
| Phép toán | Mô tả | Ví dụ | Kết quả |
| + | Cộng | x=2 x+2 | 4 |
| - | Trừ | x=2 5-x | 3 |
| \* | Nhân | x=4 x\*5 | 20 |
| / | Chia | 15/5  5/2 | 3  2.5 |
| % | Chia lấy phần dƣ | 5%2  10%8  10%2 | 1  2  0 |
| ++ | Tăng 1 đơn vị | x=5 x++ | x=6 |
| -- | Giảm 1 đơn vị | x=5 x-- | x=4 |

#### Phép gán

|  |  |  |
| --- | --- | --- |
| Phép gán | Ví dụ | Cách viết đầy đủ |
| = | 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 |
| %= | x%=y | x=x%y |

#### Các phép toán so sánh

|  |  |  |
| --- | --- | --- |
| Phép toán | Mô tả | Ví dụ |
| == | Bằng | 5==8 trả về giá trị false |
| != | Không bằng (khác) | 5!=8 trả về giá trị true |
| <> | Không bằng (khác) | 5<>8 trả về true |
| > | Lớn hơn | 5>8 trả về false |
| < | Nhỏ hơn | 5<8 trả về true |
| >= | Lớn hơn hoặc bằng | 5>=8 trả về false |
| <= | Nhỏ hơn hoặc bằng | 5<=8 trả về true |

#### Các phép toán logic

|  |  |  |
| --- | --- | --- |
| Phép toán | Mô tả | Ví dụ |
| && | And | x=6 y=3  (x < 10 && y > 1) trả về giá trị true |
| || | Or | x=6 y=3  (x==5 || y==5) trả về giá trị false |
| ! | Not | x=6 y=3  !(x==y) trả về giá trị true |

### Mệnh đề điều kiện

#### Mệnh đề if

Mệnh đề if: sử dụng mệnh đề này để thực thi một số mã lệnh chỉ khi điều kiện đưa ra là đúng.

Cú pháp:

if(điều kiện) {

// các mã lệnh thực thi khi điều kiện đúng;

}

#### Mệnh đề if…else

Mệnh đề if...else: sử dụng mệnh đề này để thực thi một số mã lệnh khi điều kiện đúng và thực thi một số mã lệnh khác khi điều kiện là sai.

Cú pháp:

if(điều kiện)

{// các mã lệnh thực thi khi điều kiện đúng}

else

{// các mã lệnh thực thi khi điều kiện sai}

#### Mệnh đề if … else if … if

Mệnh đề if ... else if … else: sử dụng mệnh đề này để chọn một trong nhiều khối mã lệnh để thực thi.

Cú pháp:

if(điều kiện)

{// các mã lệnh thực thi khi điều kiện đúng} else if(điều kiện)

{// các mã lệnh thực thi khi điều kiện đúng}

else

{// các mã lệnh thực thi khi điều kiện sai}

#### Mệnh đề swtich

Mệnh để switch: Sử dụng mệnh đề này để chọn một trong nhiều khối mã lệnh để thực thi.

Cú pháp:

switch(bien){

case gt1:

// các câu lệnh;

break; case gt2:

// các câu lệnh;

break;

…

case n:

// các câu lệnh;

break; default: // các câu lệnh;

}

### Vòng lặp

#### Vòng lặp while

While là câu lệnh lặp không biết trƣớc số lần lặp tức là nó lặp một đoạn lệnh nào đó cho đến khi thỏa mãn điều kiện đặt ra.

Cú pháp:

while(điều kiện) {

// các mã lệnh thực thi

}

Ví dụ:

|  |  |
| --- | --- |
| Ví dụ | Kết quả |
| <html>  <body>  <?php  $num=array(2,3,5,7,11,13,17,19);  $i=0; while($i<8)  {  echo $num[$i]. " ";  $i++;  }  ?>  </body>  </html> | 2 3 5 7 11 1317 19 |

#### Vòng lặp do … while

Do… while là câu lệnh lặp không biết trước số lần lặp tức là nó lặp 1 đoạn lệnh nào đó cho đến khi thỏa mãn điều kiện đặt ra. So với cấu trúc while thì cấu trúc do…while sẽ thực hiện ít nhất một lần lặp.

Cú pháp:

do

{

// các mã lệnh thực thi

}while(điều kiện);

Ví dụ:

|  |  |
| --- | --- |
| Ví dụ | Kết quả |
| <html>  <body>  <?php  $num=array(2,3,5,7,11,13,17,19);  $i=0; do{  echo $num[$i]. " ";  $i++;} while($i<8);  ?>  </body>  </html> | 2 3 5 7 11 13 17 19 |

#### Vòng lặp for

For là câu lệnh lặp biết trước số lần lặp tức là nó lặp 1 đoạn lệnh nào đó cho đến khi thỏa mãn điều kiện đặt ra.

Cú pháp:

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

// các mã lệnh thực thi

}

Trong đó:

Biểu thức 1: thường được sử dụng để khởi tạo biến đếm, biến điều khiển vòng lặp, …

Biểu thức 2: là biểu thức điều kiện, nếu biểu thức này đƣợc đánh giá là TRUE thì vòng lặp sẽ tiếp tục thực hiện, ngược lại sẽ dừng khi biểu thức được đánh giá là FALSE.

Biểu thức 3: thường được sử dụng để điều khiển vòng lặp, tăng hoặc giảm biến điều khiển, … Biểu thức này được thực thi ở cuối mỗi vòng lặp.

Lưu ý: Trong 3 biểu thức trên nếu ở mỗi biểu thức có hơn 1 biểu thức con thì giữa các biểu thức con được phân cách bởi dấu phẩy.

Ví dụ: Tính tổng n số nguyên dương đầu tiên (với n=5)

<html>

<body>

<?php

$n=5;

for($i=1, $s=0; $i<=$n;$i++)

$s+=$i;

echo "Tong cua ".$n." so nguyen duong dau tien la:".$s;

?>

</body>

</html>

#### Vòng lặp foreach

Cú pháp:

foreach($tên\_mảng as $value){

// các mã lệnh thực thi

}

Mỗi lần lặp đƣợc thực hiện thì giá trị của phần tử hiện tại trong mảng sẽ được gán bằng giá trị của $value và con trỏ của mảng dịch chuyển đi một vì vậy trong lần lặp tiếp theo sẽ là giá trị của phần tử tiếp theo trong mảng.

Ví dụ: Tính tổng các phần tử trong mảng

<html>

<body>

<?php

$num=array(2,4,6,8);

$s=0;

foreach($num as $value)

$s+=$value;

echo "Tong cua cac phan tu trong mang la: ".$s;

?>

</body>

</html>

### Mảng trong PHP

Mảng có thể chứa nhiều thông tin riêng biệt, mảng giống như một danh sách các giá trị, mỗi giá trị là một biến, một chuỗi, một số, thậm chí là một mảng khác.

Mảng được câu tạo dƣới dạng một chuỗi các cặp khóa giá trị, mỗi mục trong danh sách đều có một khóa (hoặc chỉ số index gắn liền với chúng).

Mảng trong PHP cũng phải tuân thủ theo quy tắc đặt tên của biến, để nhận giá trị từ mảng ta cần tham chiếu đến tên mảng và khóa đặt trong dấu ngoặc vuông:

Ví dụ: echo $list [2] hoặc echo $nation [„VN‟];

#### Tạo mảng

* Dùng hàm array()

Ví dụ:

<?php

$mang = array ('VN' => 'Việt Nam', 'LA' => 'Lào'); echo " VN là tên viết tắt của {$mang['VN']}";

?>

Nếu khóa đặt trước là dạng số thì các giá trị sau được tự động gán các khóa số tiếp tục.

Ví dụ:

<?php

$mang = array (1 => 'Việt Nam','Lào'); echo " LA là tên viết tắt của {$mang[2]}";

?>

* Dùng hàm range()

<?php

$mangso = range (1, 10); //cũng có thể dùng cho chữ cái echo $mangso[3]; //câu lệnh này in ra giá trị là 4, là

số đừng vị trí thứ 3 trong mảng tính từ 0

?>

* Tạo mảng với dữ liệu bất kì của vòng lặp

<?php for($i=0;$i<=100;$i++){

$mang[]=$i;

}

?>

#### Truy xuất mảng

Chúng ta có thể dùng hàm foreach() để truy xuất các phần tử mảng (hàm này làm việc dựa trên bản sao của mảng nên sẽ không ảnh hưởng đến dữ liệu của mảng nhưng sẽ ảnh hƣởng đến bộ nhớ nếu làm việc với mảng lớn):

Ví dụ:

<?php

$months = array (1 => 'January','Feburary', 'March', 'April', 'May', 'June', 'July', 'August', 'September', 'October', 'November', 'December');

$days = range (1, 31);

$years = range (2000, 2020); echo '<select name="month">';

foreach ($months as $key => $value){

echo "<option value=\"$key\">$value</option>\n";

}

echo '</select><select name="day">'; foreach ($days as $value){

echo "<option value=\"$value\">$value</option>\n";

}

echo '</select><select name="years">'; foreach ($years as $value){

echo "<option value=\"$value\">$value</option>\n";

}

echo ‗</select>‘;

?>

#### Các hàm thông dụng sử lý mảng

Hàm count(), sizeof() : trả về kích thước của mảng

Các hàm “di chuyển” trong mảng :

* current()/end() : trả về phần tử đầu tiên/ cuối cùng của mảng
* prev() / next() : trả về phần tử liền trƣớc /kế kiếp

Các hàm sắp xếp mảng:

* sort() : sắp xếp mảng theo value của phần tử.
* ksort():sắp xếp mảng theo key của phần tử.

Kiểm tra một biến có phải là mảng : is\_array() trả về 1 (true) và 0 (false)

#### Mảng đa chiều

Ví dụ:

<?php

$nations = array ('VN' => 'Việt Nam', 'US' => 'Uited States', 'UK' => 'United Kingdom');

$vietnam = array('HN' => 'Hà Nội', 'HCM' => 'Hồ Chí Minh');

$n = array ('VN' => $vietnam, 'US' => @$unitedstates, 'UK' => @$unitedkingdom);

echo " Thủ đô của {$nations['VN']} là {$n['VN']['HN']}";

// kết quả sẽ là: Thủ đô của Việt Nam là Hà Nội.

?>

#### Sắp xếp mảng

Để sắp xếp mảng chúng ta có hai sự chọn lựa: sắp xếp theo khoá hoặc sắp xếp theo giá trị.

Ví dụ:

<?php

echo "Mảng thể hiện bảng điểm</br>";

$mark = array ('10' => 'Peter', '05' => 'Tom', '02' => 'Mary');

foreach ($mark as $key => $value)

{ echo "$key\t$value </br>";} echo '</pre>';

//Xắp xếp theo thứ tự chữ cái tên

echo "</br> Xắp theo tên </br>";

asort ($mark); //Các khóa vẫn được gắn với giá trị của nó foreach ($mark as $key => $value)

{

echo "$key\t$value </br>";} echo '</pre>';

//Xắp xép theo thứ tự khóa tăng dần

echo "</br> Xắp theo thứ tự khóa tăng dần </br>";

ksort ($mark); //Các khóa vẫn được gắn với giá trị của nó foreach ($mark as $key => $value)

{

echo "$key\t$value </br>";} echo '</pre>';

//Xắp xép theo thứ tự khóa giảm dần

echo "</br> Xắp theo thứ tự khóa giảm dần </br>";

krsort ($mark); //Các khóa vẫn được gắn với giá trị của nó foreach ($mark as $key => $value)

{

echo "$key\t$value </br>";} echo '</pre>';

?>

### Hàm trong PHP

Cú pháp:

function tên\_hàm([các\_tham\_số]) {

// Khai báo biến riêng

// Câu lệnh PHP

[ return giá\_trị; ]

}

Sử dụng hàm:

Hàm không có giá trị trả về : tên\_hàm([các\_tham\_số]);

Hàm có giá trị trả về : tham gia vào các biểu thức, gán giá trị cho biến.

Ví dụ: : hàm không có giá trị trả về

<?php

$arr = array(hoten=>”Hà Giang”,diachi=>”Đà Nẵng”,tuoi=>15,phai=>”Nữ”);

function display($a){

echo “<table border=1>”;

reset($a); // trỏ về phần tử đầu tiên của mảng $a;

$value=current($a); // phần tử hiện thời của mảng $a

$str=””; while($value){

$str**.**=”<tr><td>”.$value.”</td></tr>”;

$value=next($a); // phần tử kế tiếp của mảng $a

}

$str.= “</table>”; echo $str;

}

display($arr);

?>

Ví dụ: : hàm có giá trị trả về

<?php

$arr=array(hoten=>"Hà Giang",diachi=>"Đà Nắng"); function is\_hoten($mang){

// trả về value của phần tử có **key** là **hoten**

while(list($k,$v)=each($mang))

{ if($k=="hoten"){

return $v;exit;

}

}

}

echo is\_hoten($arr);

?>

#### Một số hàm thông dụng trong PHP

* Hàm số toán học

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên hàm** | **Cú pháp** | **Chức năng** | **Ví dụ** | **Kết quả** |
| abs | abs($number) | Trả về giá trị tuyệt đối của $number, nếu  $number là float thì kết quả là float ngƣợc lại là interger | $abs=abs(-4.2);  $abs=$abs(-4); | $abs = 4.2  $abs=4 |
| ciel | ciel($float) | Làm tròn lên số $float | $a=ciel(5.1); | $a=6 |
| floor | floor($float) | Làm tròn xuống của  $float | $a=floor(5.9);  $a=floor(-3.51); | $a=5  $a= - 4 |
| max | max($num1,$nu m2,..,$numn) | Tìm số lớn nhất | $ln=max(1,5,9,4); | $ln=9 |
| min | min($num1,$nu m2,…,$numn) | Tìm số nhỏ nhất | $nn=min(1,5,9,4); | $nn=1; |
| pi | pi() | Lấy giá trị số pi | echo pi(); | 3.1415926535898 |
| pow | pow($var,$exp) | Trả về giá trị của phép toán $var mũ $exp; | echo pow(2,8); | 246 |
| rand | rand($min,$max  ) | Lấy số nguyên ngẫu nhiên có giá trị $min  <rand() <$max | echo rand(1,10); | Kết quả trên màn hình sẽ là 1 số ngẫu nhiên từ 1 đến 10 |
| round | round($val,$pre cision) | Làm tròn số $val với  $precision số lẻ | echo round(1.2434,2); | 1.24 |
| sqrt | sqrt($var) | Phép toán lấy căn bậc 2 của $var | echo sqrt(9); | 3 |

* Hàm xử lý chuỗi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Tên hàm** | **Cú pháp** | **Chức năng** | **Ví dụ** | **Kết quả** |
| strlen | strlen($str) | Đếm tổng số ký tự có trong $str | $str=”chao”;  echo strlen($str); | 3 |
| str\_word\_count | str\_word\_count($str) | Đếm tổng số từ có trong $str | $str=”mot hai ba”;  echo str\_word\_count($st r); | 3 |
| addslashes | addslashes($str) | Thêm vào ký tự \ (back slash) trƣớc các ký tự „ và “ trong một chuỗi, có tác dụng tránh lỗi SQL injection khi thêm dữ liệu vào cơ sở dữ liệu | echo addslashes('chao ban "moi"'); | chao ban  \"moi\" |
| **Tên hàm** | **Cú pháp** | **Chức năng** | **Ví dụ** | **Kết quả** |
| stripslashes | stripslashes($str) | Loại bỏ các ký tự  \ trong chuỗi. | $str=‟chao ban  \”moi\”‟;  Stripslashes($str); | chao ban “moi” |
| str\_repeat | str\_repeat($str,$n) | Lặp lại chuỗi $str theo $n lần | echo str\_repeat(„chao‟,2)  ; | Chaoch ao |
| str\_replace | str\_replace($chuoi\_t im,$chuoi\_thay\_the,  $chuoi\_nguon) | Tìm kiếm và thay thế chuỗi | $str=”chao ban”;  $str=str\_replace(„ba n‟,‟Tam‟,$str);  echo $str; | chao Tam |

### Sử dụng lại mã PHP

Sử dụng hàm include(), require() để chèn tệp tin PHP, HTML vào một trang PHP khác. Ngoài ra còn có: include\_once(), require\_once(), tuy nhiên một số lập trình viên khuyên không lạm dụng, vì làm chậm thực thi script.

Phân biệt include() và require()

* include(“tên\_tệp”) : nếu tên\_tệp không tồn tại thì PHP thông báo lỗi và vẫn *tiếp tục* thực thi script còn lại.
* require(“tên\_tệp”) : nếu tên\_tệp không tồn tại thì PHP thông báo lỗi và *dừng* thực thi script còn lại.

Sử dụng include(), require() là một cách tái sử dụng đoạn mã đã viết.

### Lớp đối tượng - class

* Cách khai báo:

class tên\_lớp(){

var $tên\_biến; function tên\_hàm(){

$this->$tên\_biến=giá\_trị;

}

}

* Cách sử dụng:

$bien = new tên\_lớp();

Ví dụ:

class counter()

{

var $count=0; function increment()

{

$this->count++;

}

}

$a=new counter(); // khởi tạo

$a->increment(); // gọi hàm

echo $a->count; // kết quả là 1;

### Xử lý form trong PHP

Một trong những ứng dụng quan trọng của PHP đó là giúp tương tác xử lý dữ liệu trên form của người sử dụng. Nhằm mục đích giúp cập nhật thông tin một cách linh động và dễ dàng quản lý chung hơn bởi sự kết hợp tuyệt vời của cơ sở dữ liệu. Tuy nhiên để làm được điều ấy PHP yêu cầu form phải đáp ứng 1 số quy định chung đặt ra.



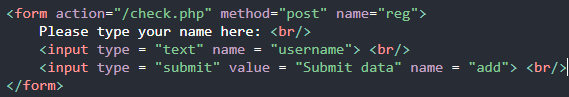
Hình : thẻ <form>

Chúng ta thấy rằng 1 form phải bao gồm:

**Name:** Tên form để dễ dàng tách biệt với giá trị của chúng.

**Action:** hành động chuyển tiếp đến link xử lý.

**Method:** Là phương thức truyền bao gồm POST và GET.



hình: ví dụ về thẻ <form>

Như vậy ta thấy rằng. Đoạn code trên làm những việc sau.

Đầu tiên khi khách nhập liệu username vào thì chúng sẽ chuyển tới trang check.php để tiến hành xử lý thông tin. Trên phương thức POST, với tên form là reg. Giá trị mà chúng ta gởi là username vừa nhập liệu.

PHP cho phép ta lấy giá trị dựa vào 2 phƣơng thức POST và GET. Đối với **POST** ta có : **$\_POST[‘Giá trị’]**

Đối với **GET** ta có : **$\_GET[‘Giá trị’]**

Vậy với đoạn code trên có thể lấy được biến xử lý là : $\_POST[„username‟]; Username là tên của field mà người sử dụng nhập liệu vào.

#### Phương thức GET

Dữ liệu gửi từ trình duyệt lên qua phương thức GET là phần dữ liệu được nhập trực tiếp theo sau địa chỉ URL do trình duyệt gửi lên, được phân biệt với tên file script bằng dấu hỏi chấm (?).

Khi trình duyệt gửi các thông tin này lên máy chủ, PHP sẽ tự động sinh ra một mảng có tên là $\_GET[] để nắm giữ tất cả các cặp biến và giá trị đó, trong đó, chỉ số của mảng chính là một chuỗi mang tên của tên biến và giá trị của chỉ số đó chính là giá trị của biến do trình duyệt gửi lên.

#### Phương thức POST

Đối với những form sử dụng phƣơng thức post (method=‟post‟) thì dữ liệu trên form sẽ không hiển thị trên ô địa chỉ. Do đó để lấy giá trị trên form ta sử dụng $\_POST.

## Khái niệm cơ bản về cookie và session trong PHP

Cookie và session là hai phương pháp sử dụng để quản lý các phiên làm việc giữa người sử dụng và hệ thống. Việc quản lý phiên làm việc này sẽ giúp bạn tạo ra sự chứng thực hiệu quả bởi việc xác nhận thông tin trƣớc khi truy cập vào một phân vùng cố định. Ngoài ra, việc quản lý tốt phiên làm việc cũng giúp ngƣời truy cập cảm thấy dễ dàng sử dụng dịch vụ của trang web cho những lần truy cập sau. Bởi cớ chế quản lý phiên làm việc ghi nhận lại quá trình truy cập của người sử dụng khi họ thăm viếc trang web của bạn lần đầu.

### Tổng quan về cookie

Cookie là 1 đoạn dữ liệu được ghi vào đĩa cứng hoặc bộ nhớ của máy người sử dụng. Nó được trình duyệt gởi ngược lên lại server mỗi khi browser tải 1 trang web từ server. Những thông tin được lưu trữ trong cookie hoàn toàn phụ thuộc vào website trên server. Mỗi website có thể lƣu trữ những thông tin khác nhau trong cookie, ví dụ thời điểm lần cuối ta ghé thăm website, đánh dấu ta đã login hay chưa, v.v...

Cookie được tạo ra bởi website và gởi tới browser, do vậy 2 website khác nhau (cho dù cùng host trên 1 server) sẽ có 2 cookie khác nhau gởi tới browser. Ngoài ra, mỗi browser quản lý và lưu trữ cookie theo cách riêng của mình, cho nên 2 browser cùng truy cập vào 1 website sẽ nhận được 2 cookie khác nhau.

Các cookie được sử dụng trong các dịch vụ thương mại điện tử để hỗ trợ chức năng mua hàng trực tuyến, máy chủ có thể theo dõi khách hàng và sao lưu các giao dịch của họ khi họ di chuyển trong site.

Mặc dù bản thân cookie thường trú rất hữu ích và thuận tiện nhưng kèm theo đó là những rủi ro có liên quan được lưu trữ.

Cookie ảnh hưởng tới sự riêng tƣ của người dùng. cũng như rò rỉ thông tin cá nhân . Cookie theo dõi người dùng đã ghé thăm những nơi nào và đã xem những gì trên web. Bản thân các cookie không thể dùng để phát tán virus, mã độc. Tuy nhiên nó có thể thu thập khá nhiều thông tin cá nhân của bạn nhất là những thông tin bạn cung cấp trên trang web như thông tin thẻ tín dụng… nên cookie có thể làm tăng nguy cơ mất thông tin đăng nhập nếu như ngƣời khác sử dụng máy tính của bạn, hoặc trường hợp máy tính của bạn bị xâm nhập, đánh cắp.

#### Thiết lặp cookie

Để thiết lập cookie ta sử dụng cú pháp: Setcookie("tên cookie","giá trị", thời gian sống)

Tên cookie là tên mà chúng ta đặt cho phiên làm việc.

Giá trị là thông số của tên cookie.

Ví dụ:

Setcookie("username","admin", time() +3600)

Nhƣ ví dụ trên ta thấy với tên là username và giá trị là admin, có thời gian sống là 1 giờ tính từ thời điểm thiết lập.

Chú ý: Kịch bản cookie phải đặt trên mọi giá trị trả về bao gồm thẻ HTML và lệnh echo.

Kịch bản cookie phải đặt trên mọi giá trị trả về bao gồm thẻ HTML và lệnh echo.

#### Sử dụng cookie

Để sử dụng lại cookie vừa thiết lập, chúng ta sử dụng cú pháp:

Cú pháp:

$\_COOKIE["tên cookies"]

Tên cookie là tên mà chúng ta thiết lập phía trên.

Ví dụ: Tạo ra 2 trang (**cookie.php** dùng để khởi tạo cookie có tên là name, nội dung là Nguyen Van A và có thời gian sống là 1 giờ tính từ thời điểm thiết lập) và (**cookie2.php** dùng để hiển thị nội dung của cookie có tên là name đã đƣợc khởi tạo ở trang **cookie.php**)

Cách thực hiện:

Nội dung trang **cookie.php**:

<?php

setcookie("name","Nguyen Van A",time() + 3600);

?>

<html>

<head>

<title>Test page 1</title></head>

<body>

<b><a href=cookie2.php>Click here</a></b>

</body>

</html>

Tiếp tục tạo trang **cookie2.php** với nội dung sau

<html>

<head><title>Result Page</title></head>

<body>

<?php

echo "Ten cua ban la <b>".$\_COOKIE['name']."</b>";

?>

</body>

</html>

#### Hủy cookie

Để hủy 1 cookie đã đƣợc tạo ta có thể dùng 1 trong 2 cách sau:

Cách 1:

Gọi hàm setcookie với chỉ duy nhất tên cookie mà thôi.

Cú pháp:

setcookie("Tên cookie")

Cách 2: Dùng thời gian hết hạn cookie là thời điểm trong quá khứ.

Ví dụ:

setcookie("name","Kenny Huy",time()-3600);

### Tổng quan về session

Một cách khác quản lý người sử dụng là session. Session được hiểu là khoảng thời gian người sử dụng giao tiếp với 1 ứng dụng. Một session đƣợc bắt đầu khi người sử dụng truy cập vào ứng dụng lần đầu tiên, và kết thúc khi ngƣời sử dụng thoát khỏi ứng dụng. Mỗi session sẽ có được cấp một định danh (ID) khác nhau và nội dung được lưu trong thư mục thiết lập trong file php.ini (tham số session.save\_path).

#### Thiết lặp session

Để thiết lập 1 session ta sử dụng cú pháp: session\_start(). Đoạn code này phải được nằm trên các kịch bản HTML hoặc những lệnh echo, printf.

Để thiết lập 1 giá trị session, ngoài việc cho phép bắt đầu thực thi session. Chúng ta còn phải đăng ký 1 giá trị session. Để tiện cho việc gán giá trị cho session đó.

Cú pháp:

session\_register("Name")

Ví dụ:

<?php

session\_start(); session\_register("username");

?>

#### Cách sử dụng

Để sử dụng giá trị của session ta sử dụng mã lệnh sau:

Cú pháp:

$\_SESSION["name"]

Với Name là tên mà chúng ta sử dụng hàm session\_register("name") để khai báo.

Ví dụ: Tạo trang **session.php** với nội dung sau:

<?php

session\_start(); session\_register("name");

$\_SESSION["name"] = "TEST NAME";

?>

<html>

<head>

<title>Test page 1</title></head>

<body>

<b><a href=session2.php>Click here</a></b>

</body>

</html>

#### Hủy session

Để hủy bỏ giá trị của session ta có thể huỷ bỏ từng session đã đăng ký hoặc huỷ bỏ toàn bộ các sesseion đã đăng ký.

Để huỷ bỏ tất cả các session đã đăng ký ta sử dụng lệnh: session\_destroy()

Huỷ bỏ 1 session đã đăng ký ta sử dụng lệnh: session\_unset(„tên\_session‟)

Ví dụ: Tạo trang **session3.php** với nội dung sau:

<?php

session\_start(); session\_destroy();

?>

<html>

<head>

<title>Test page 1</title></head>

<body>

<b><a href=session2.php>Click here</a></b>

</body>

</html>

# CHƯƠNG 4: PHP VÀ MYSQL

## Giới thiệu

Cơ sở dữ liệu MySQL là tập hợp các đối tượng: bảng, bảng ảo.. cho phép người dùng lưu trữ và truy xuất các thông tin đã được tổ chức và lưu trữ bên trong đó.

## Đặc điểm

Sử dụng cho các ứng dụng Web có quy mô vừa và nhỏ. Để thực hiện các thao tác trên cơ sở dữ liệu MySQL, có thể sử dụng giao diện đồ họa hay dùng dòng lệnh (command line). Các tập tin vật lý lƣu trữ cơ sở dữ liệu. Mỗi bảng sẽ được lưu trữ dưới ba tập tin vật lý.

.frm: lưu định dạng (cấu trúc) của bảng

.MYD: lưu nội dung của bảng

.MYI: lưu chỉ mục của bảng

* Quy tắc đặt tên:
* Tên không kết thúc bằng khoảng trắng
* Tên cơ sở dữ liệu không có các ký tự: /, \, ., :, \*, „‟, <, >
* Tên bảng không có các ký tự: /,\,.,:,\*,**‘’, <, >, |**
* Chiều dài của tên tối đa là 64 ký tự không dấu. Khi sử dụng các ký tự đa byte thì chiều dài sẽ dựa trên tổng số byte của tất cả các ký tự được dùng.

## Thao tác với Cơ sở dữ liệu

### Tạo Cơ sở dữ liệu

Có hai cách để tạo một Cơ sở dữ liệu: dùng giao diện đồ họa hoặc dùng dòng lệnh.

* **Tạo Cơ sở dữ liệu sử dụng giao diện**

Bước 1: Khởi động phpmyadmin

Bước 2: Nhập tên cơ sở dữ liệu muốn tạo vào mục Create New Database và chọn các thông tin khác

Bước 3: Nhấn “Create” để hoàn thành việc tạo cơ sở dữ liệu

* **Tạo Cơ sở dữ liệu sử dụng câu lệnh CREATE DATABASE**

Cú pháp:

CREATE DATABASE Tên\_CSDL

[[ DEFAULT] CHARACTER SET <character set name>]

[[ DEFAULT] COLLATE <collation name>]

Trong đó:

CHARACTER SET: xác định bộ ký tự mặc định cho CSDL mới COLLATE: xác định bộ collation

Character set name: tên của một bộ mã bao gồm các ký tự, ký số, và biểu tượng để lưu trữ các thông tin trong CSDL

Collation name: tên của bộ mã tùy theo từng khu vực dựa trên bộ mã chuẩn character set name

Ví dụ:

CREATE DATABASE QLSV CHARACTER SET utf-8 COLLATE

utf8\_unicode\_ci

Hay

CREATE DATADASE QLSV

### Xóa Cơ sở dữ liệu

Có hai cách xóa cơ sở dữ liệu: Dùng giao diện đồ hoạ hoặc dùng dòng lệnh.

* **Xóa Cơ sở dữ liệu sử dụng giao diện**

Bước 1: Chọn Cơ sở dữ liệu cần xóa, sau đó nhấn DROP

Bước 2: Xác nhận việc xóa Cơ sở dữ liệu đã chọn, nhấn OK để xóa

* **Dùng câu lệnh DROP DATABASE**

Cú pháp:

DROP DATABASE TEN\_CSDL

Ví dụ: Xóa CSDL QLSV

DROP DATABASE QLSV

## Thao tác với bảng

Cú pháp:

Create table ten\_bang (

Ten\_cot\_1 kieu\_du\_lieu[(kich\_co)][Not NULL],

Ten\_cot\_2 kieu\_du\_lieu[(kich\_co)][Not NULL],

…

)

Chú ý: NOT NULL có nghĩa là không cho phép để trống dữ liệu trong cột

Ví dụ: tạo bảng hang\_sua với hai cột dữ liệu không đƣợc phép bỏ trống là Ma\_hang\_sua và Ten\_hang\_sua

CREATE TABLE hang\_sua (

Ma\_hang\_sua varchar(20) NOT NULL,

Ten\_hang\_sua varchar(100) NOT NULL,

Dia\_chi varchar(200),

Dien\_thoai varchar(20),

Email varchar(100)

)

Cách tạo bảng thiết lập khoá chính (primary key)

Cú pháp: primary key (cac\_cot\_tham\_gia\_lam\_khoa\_chinh).

Chú ý: Đối với bảng mà khoá chính chỉ có một cột thì có thể khai báo khoá chính ngay sau khi khai báo cột. Đối với bảng có nhiều cột tham gia làm khoá thì thiết lập các cột trước rồi mới tạo khoá chính sau.

Ví dụ: tạo bảng hang\_sua như trên với khoá chính là cột Ma\_hang\_sua

CREATE TABLE hang\_sua (

Ma\_hang\_sua varchar(20) NOT NULL PRIMARY KEY,

Ten\_hang\_sua varchar(100) NOT NULL,

Dia\_chi varchar(200),

Dien\_thoai varchar(20),

Email varchar(100)

)

### Thay đổi cấu trúc bảng

Có thể thay đổi lại cấu trúc bảng mà không làm mất đi dữ liệu đã có trong bảng bằng cách sử dụng câu lệnh ALTER TABLE.

* **Thêm cột mới trong bảng**

Cú pháp:

ALTER TABLE Ten\_bang ADD

Ten\_cot kieu\_du\_lieu[(kich\_co)][…]

Chú ý: tên cột mới thêm vào phải khác với tên cột đã có trong bảng

* **Sửa đổi kiểu dữ liệu của cột**

Cú pháp:

ALTER TABLE ten\_bang

CHANGE ten\_cot\_cu ten\_cot\_moi kieu\_du\_lieu\_moi [kich\_co]

Chú ý: câu lệnh này có thể dùng để thay đổi tên của cột thành tên mới.

### Hủy cột trong bảng

Khi không cần sử dụng cột trong bảng, chúng ta có thể sử dụng cú pháp ALTER TABLE để huỷ bỏ cột.

Cú pháp:

ALTER TABLE Ten\_bang

DROP COLUMN Ten\_cot,..

### Xóa bảng

Để xoá bảng trong cơ sở dữ liệu, ta sử dụng lệnh DROP TABLE

Cú pháp:

DROP TABLE Ten\_bang

## Import và export dữ liệu

Trong quá trình làm việc với cơ sở dữ liệu, việc sao lưu và phục hồi (import / export) dữ liệu là hết sức quan trọng. Chúng ta có thể di chuyển dữ liệu từ máy client lên máy server hoặc ngƣời lại nhờ vào hai quá bước này. Export là để sao lưu dữ liệu lại còn Import là dùng để phục hồi dữ liệu.

### Import dữ liệu

Nhập dữ liệu từ bên ngoài vào Database trong MySQL

Bước 1: Vào Database muốn import dữ liệu

Bước 2: Chọn chức năng import

Bước 3: Chọn tập tin chứa dữ liệu cần import ở ô *Location of the text file*

Bước 4: Xác nhận (nhấn Go) để hoàn thành việc import dữ liệu

### Export dữ liệu

Xuất dữ liệu từ Database trong MySQL ra tập tin có dạng **\*.sql**:

Bước 1: Chọn Database muốn export dữ liệu

Bước 2: Chọn chức năng export

Bước 3: Chọn các bảng trong Database cần export

Bước 4: Chọn loại file để export

Bước 5: Thiết lập các thuộc tính cho file export, chọn *Save as file*

Bước 6: Đặt tên cho file export (nhấn Go để qua phần chọn nơi lƣu trữ)

Bước 7: Chọn nơi lưu trữ file export

Bước 8: Click “Save” để hoàn thành việc export dữ liệu

## Kết nối Cơ sở dữ liệu MySql với PHP

### Các bước xây dựng chương trình có kết nối Cơ sở dữ liệu

Thông thường, trong một ứng dụng có giao tiếp với Cơ sở dữ liệu, ta phải làm theo bốn trình tự sau:

Bước 1: Thiết lập kết nối tới cơ sở dữ liệu.

Bước 2: Lựa chọn Cơ sở dữ liệu.

Bước 3: Tiến hành các truy vấn SQL, xử lý các kết quả trả về nếu có

Bước 4: Đóng kết nối tới Cơ sở dữ liệu.

Nếu như trong lập trình thông thƣờng trên Windows sử dụng các chƣơng trình điều khiển trung gian (ADO, ODBC...) để thực hiện kết nối và truy vấn, thì trong PHP, khi lập trình tương tác với cơ sở dữ liệu, chúng ta thường sử dụng thông qua các hàm.

**Chú ý:** Trước khi sử dụng PHP để lập trình với MySQL, hãy sử dụng chương trình quản lý phpMyAdmin để tạo trước một cơ sở dữ liệu, vài bảng cũng như người dùng.

* **Thiết lập kết nối tới MySQL**

Cú pháp:

mysql\_connect(servername,username,password);

Trong đó:

*servername*: tham số tùy chọn, xác định server cần phải kết nối tới. Giá trị phục thuộc vào nhà cung cấp host. Khi chúng ta thử nghiệm với việc cài đặt xampp hay bất kỳ webserver nào khác thì servername mặc định ở đây là “localhost”.

*username*: tham số tùy chọn, xác định tên người dùng đăng nhập vào hệ thống. Giá trị mặc định là tên của người dùng làm chủ tiến trình của server.

*password*: tham số tùy chọn, xác định mật khẩu của người dùng.

Ví dụ:

<?php

mysql\_connect ("localhost", "root","");

?>

Khi kết nối tới MySQL thành công, hàm sẽ trả về giá trị là một số nguyên định danh của liên kết, ngược lại, hàm trả về giá trị false.

Ta có thể dùng hàm if để kiểm tra xem có kết nối được tới MySQL hay không.

Ví dụ:

<?php

$link = mysql\_connect("localhost", "mysql\_user", "mysql\_password");

if (!$link)

{

echo "Không thể kết nối được tới CSDL MySQL";

}

?>

Đoạn code bên trên có thể được viết ngắn gọn như sau:

<?php

$link = mysql\_connect("localhost", "mysql\_user", "mysql\_password") or die ("Không thể kết nối được tới CSDL MySQL");

?>

* **Lựa chọn Cơ sở dữ liệu**

Để lựa chọn một CSDL (cơ sở dữ liệu) nào đó mà người sử dụng có tên là tên\_truy\_cập" có quyền sử dụng, ta dùng hàm mysql\_select\_db

Cú pháp:

mysql\_select\_db (tên\_CSDL);

Hàm này chỉ được dùng sau khi thiết lập kết nối bằng hàm mysql\_connect. Hàm này trả về TRUE nếu thành công, FALSE nếu thất bại.

Ví dụ:

<?php

$link = mysql\_connect("localhost", "mysql\_user", "mysql\_password");

if (!$link)

{

echo "Không thể kết nối được tới MySQL";

}

if (!mysql\_select\_db ("Tên\_CSDL"))

{

echo "Không thể lựa chọn được CSDL";

}

?>

Đoạn code trên có thể viết lại như sau:

<?php

$link = mysql\_connect("localhost", "mysql\_user", "mysql\_password") or die("Không thể kết nối được tới MySQL");

mysql\_select\_db ("Tên\_CSDL") or die("Không thể lựa chọn được CSDL");

?>

* **Đóng kết nối Cơ sở dữ liệu**

Để đóng kết nối tới CSDL, ta dùng hàm mysql\_close.

Hàm này có nhiệm vụ đóng kết nối tới CSDL có mã định danh được tạo ra bởi hàm mysql\_connect().

Cú pháp:

mysql\_close(connection);

Trong đó

connection: tham số tùy chọn, xác định kết nối. Nếu không xác định thì kết nối cuối cùng được mở bởi hàm mysql\_connect() hoặc hàm mysql\_pconnect() sẽ được sử dụng.

Kết quả trả về là TRUE nếu đóng kết nối thành công, ngược lại sẽ trả về FALSE nếu thất bại.

Ví dụ:

<?php

$link = mysql\_connect("localhost", "mysql\_user", "mysql\_password") or die("Không thể kết nối được tới MySQL");

mysql\_close($link);

?>

### Thông báo lỗi

Trong quá trình làm việc với CSDL lỗi có thể phát sinh cần thông báo những lỗi phát sinh này bằng cách sử dụng hàm **mysql\_error()**.

Cú pháp:

mysql\_error(connection);

Kết quả trả về là câu thông báo lỗi nếu có lỗi phát sinh, ngược lại kết quả trả về sẽ là một chuỗi rỗng “”.

Kết hợp hàm mysql\_error() với hàm die() hoặc hàm exit() để vừa thông báo lỗi vừa kết thúc công việc

Ví dụ: thông báo lỗi nếu không thể tạo kết nối

<?php

$db = mysql\_connect("localhost","root","") or die('Không thể kết nối: ' . mysql\_error());

?>

## Thao tác trên Cơ sở dữ liêụ MySQL với PHP

### Đếm số lượng mẩu tin

Cú pháp:

mysql\_num\_rows(data);

Trong đó:

*data*: là tham số bắt buộc. Xác định con trỏ dữ liệu. Con trỏ dữ liệu là kết quả trả về của hàm mysql\_query().

Hàm này có kết quả trả về là số lượng mẩu tin nếu thành công, ngược lại kết quả trả về sẽ là FALSE nếu thất bại.

Ví dụ: đếm số lượng mẩu tin có trong bảng hang\_sua

$sql= "SELECT \* FROM hang\_sua";

$result = mysql\_query($sql);

echo mysql\_num\_rows($result);

### Hiển thị dữ liệu

Sử dụng một trong các cách sau:

Duyệt dữ liệu theo dạng mỗi mẩu tin là một dòng

Duyệt theo dạng mỗi mẫu tin là một mảng

Duyệt theo dạng mỗi mẩu tin là một đối tượng

* **Duyệt dữ liệu theo dạng mỗi mẩu tin là một dòng *mysql\_fetch\_row()***

Cú pháp:

mysql\_fetch\_row(data);

Trong đó:

*data*: là tham số bắt buộc. Xác định con trỏ dữ liệu. Con trỏ dữ liệu là kết quả trả về của hàm mysql\_query()

Kết quả trả về một mảng (có chỉ số) chứa giá trị của một dòng dữ liệu với mỗi phần tử là nội dung của một cột  truy cập bằng cách gọi từng phần tử của mảng $row[0], $row[1],$row[2],…

Ví dụ:

Duyệt dữ liệu theo dạng mỗi mẩu tin là một dòng mysql\_fetch\_row()

mysql\_select\_db("ql\_ban\_sua");

$result = mysql\_query("SELECT \* FROM KHACH\_HANG");

if (mysql\_num\_rows($result)<>0) {

print\_r(mysql\_fetch\_row($result));

}

mysql\_close($db);

Ví dụ:

Duyệt tất cả các mẩu tin: kết hợp cấu trúc lặp while và hàm mysql\_fetch\_row() đặt trong cấu trúc lặp while

if (mysql\_num\_rows($result)<>0)

{

while($row = mysql\_fetch\_row($result))

{ print\_r($row); }

}

* **Duyệt dữ liệu theo dạng mỗi mẩu tin là một mảng mysql\_fetch\_array()**

Cú pháp:

mysql\_fetch\_array(data);

Trong đó:

*data*: là tham số bắt buộc. Xác định con trỏ dữ liệu. Con trỏ dữ liệu là kết quả trả về của hàm *mysql\_query()*

Kết quả trả về là một mảng (có chỉ số chuỗi) chứa giá trị của một dòng dữ liệu với mỗi phần tử là nội dung của một cột  truy cập bằng cách gọi từng phần tử của mảng: $row[“tên cột 1”], $row[“tên cột 2”], $row[“tên cột 3”],…

Ví dụ:

if(mysql\_num\_rows($result)<>0){

echo (mysql\_fetch\_array($result));

}

Ví dụ: Duyệt tất cả các mẩu tin: kết hợp cấu trúc lặp while và hàm *mysql\_fetch\_array()* đặt trong cấu trúc lặp while.

if (mysql\_num\_rows($result)<>0){

while($array = mysql\_fetch\_array($result)) {

echo ($array);

}

}

* **Duyệt dữ liệu theo dạng mỗi mẩu tin là một đối tượng mysql\_fetch\_object()**

Cú pháp:

mysql\_fetch\_object(data)

Trong đó:

*data*: là tham số bắt buộc. Xác định con trỏ dữ liệu. Con trỏ dữ liệu là kết quả trả về của hàm *mysql\_query()*

Kết quả trả về là một mẩu tin trong bộ các mẩu tin như là một đối tượng  truy cập bằng cách gọi từng thuộc tính của đối tượng: $tên\_đối\_tƣợng  tên\_cột\_1, $tên\_đối\_tượng  tên\_cột\_2,…

Ví dụ:

if (mysql\_num\_rows($result)<>0) {

print\_r(mysql\_fetch\_object($result));

}

Ví dụ: Duyệt tất cả các mẩu tin: kết hợp cấu trúc lặp while và hàm *mysql\_fetch\_object()* đặt trong cấu trúc lặp while.

if (mysql\_num\_rows($result)<>0){

while($object = mysql\_fetch\_object($result)) { print\_r($object); }

}

### Lưu thông tin mới vào Cơ sở dữ liệu

Sử dụng hàm mysql\_query() kết hợp với câu lệnh truy vấn INSERT INTO.

Hàm này được dùng để gửi một truy vấn (hiển thị thông tin, thêm mới, xoá, cập nhật) tới một kết nối MySQL.

Ví dụ: Thêm mới một hãng sữa với giá trị cụ thể vào bảng hãng sữa đã có.

$db = mysql\_connect("localhost","root","");

if (!$db)

{

die(“Không thể kết nối CSDL: “ . mysql\_error());

}

mysql\_select\_db("ql\_ban\_sua", $db);

mysql\_query("INSERT INTO hang\_sua VALUES ('XO', 'XO', 'Khu công nghiệp Đồng Nai', ‘7456321‘, [‘xo@xo.com‘)");](mailto:xo@xo.com) mysql\_close($db);

### Cập nhật dữ liệu

Sử dụng hàm mysql\_query() kết hợp với câu lệnh truy vấn UPDATE.

Hàm này đƣợc dùng để gửi một truy vấn (hiển thị thông tin, thêm mới, xoá, cập nhật) tới một kết nối MySQL.

Ví dụ: cập nhật lại địa chỉ cho hãng sữa XO là “Khu công nghiệp Bình Dương”

$db = mysql\_connect("localhost","root","");

if (!$db){

die(“Không thể kết nối CSDL: “ . mysql\_error());

}

mysql\_select\_db("ql\_ban\_sua", $db);

mysql\_query("UPDATE hang\_sua SET Dia\_chi = 'Khu công nghiệp Bình Dương' WHERE Ma\_hang\_sua = ‘XO‘");

mysql\_close($db);

### Xóa dữ liệu

Sử dụng hàm mysql\_query() kết hợp với câu lệnh truy vấn DELETE.

Hàm này được dùng để gửi một truy vấn (hiển thị thông tin, thêm mới, xoá, cập nhật) tới một kết nối MySQL

Ví dụ: xoá thông tin của hãng sữa XO

$db = mysql\_connect("localhost","root","");

if (!$db){

die(“Không thể kết nối: “ . mysql\_error());

}

mysql\_select\_db("ql\_ban\_sua", $con);

mysql\_query("DELETE FROM hang\_sua WHERE Ma\_hang\_sua='XO'");

mysql\_close($db);

Lưu ý: PHP có thể kết hợp với các CSDL khác

PHP có thể thực thi đƣợc trên cả hai môi trường là Windows và Linux do đó PHP có thể kết nối nhiều loại CSDL (MS Access, SQL Server, …) trên hai môi trường này.

Dùng ODBC để kết nối tới nguồn dữ liệu. Với một kết nối ODBC, có thể kết nối đến nhiều CSDL khác nhau.

# DANH MỤC TÀI LIỆU THAM KHẢO