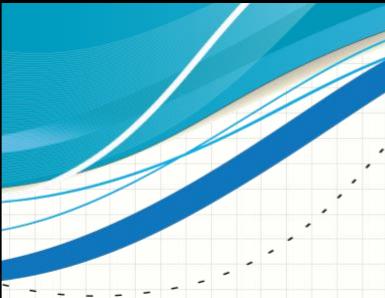


# PHÁT TRIỂN PHẦN MỀM MÃ NGUỒN MỞ

ThS. Lê Thị Bích Hằng  
[ltbhangnt@yahoo.com](mailto:ltbhangnt@yahoo.com)



## ***PHẦN III – LẬP TRÌNH PHP***

## NỘI DUNG HỌC PHẦN

- Môi trường thực hành:
  - Dùng Macro Media DreamWeaver/Eclipse/Net Bean/PHP Designer/Notepad++...
  - Chạy trên WAMP/XAMPP

3

## MỤC TIÊU HỌC PHẦN

- Nắm vững khái niệm cơ sở trong ngôn ngữ lập trình PHP
- Sử dụng những điều khiển cơ sở trong thiết kế Web với PHP
- Xây dựng và xử lý thành phần giao diện của ứng dụng

4



## BÀI 1

# GIỚI THIỆU PHP

- Các khái niệm cơ bản
- Giới thiệu PHP
- Cơ chế hoạt động của Web Server
- Giới thiệu WAMPP Server
- Công cụ để xây dựng ứng dụng PHP



### Các khái niệm cơ bản

- HTML (Hypertext Markup Language)
- Web Programming Language
- Web Server/ Web Client
- Database Server
- Web Browser
- URL (Uniform Resource Locator)
- HTTP (Hypertext Transfer Protocol)

6

## HTML

- HTML (HyperText Markup Language)
- Ngôn ngữ dùng để xây dựng các trang web.
- Gồm các thẻ (tag) giúp trình duyệt web (web browser) biết cách định dạng thông tin hiển thị.

**Ví dụ:** nội dung trang web Chao.html

```
<html>
  <head><title>Chào mừng</title></head>
  <body>
    <h1 align="center">Chào mừng các bạn đến với ngôn ngữ HTML</h1>
  </body>
</html>
```

7

## Web Programming Language

- Là ngôn ngữ lập trình được sử dụng để hỗ trợ và tăng cường các khả năng của các ứng dụng web.
- Một số ngôn ngữ lập trình web thường được dùng:
  - Xử lý ở Client: Javascript, VBScript
  - Xử lý ở Server: ASP, PHP, JSP, ...

8

## Web Server và Web Client

- **Web Server**

- Máy tính lưu trữ website.
- Trả kết quả về cho Web Client khi nhận được yêu cầu.
- Tất cả các Web Server đều hiểu và chạy được các file **\*.htm** và **\*.html**, tuy nhiên mỗi Web Server lại phục vụ một số kiểu file chuyên biệt chẳng hạn như IIS dành cho **\*.asp**, **\*.aspx...**; Apache dành cho **\*.php...**; Sun Java System Web Server dành cho **\*.jsp...**

- **Web Client**

- Máy tính dùng để truy cập các trang web.
- Có khả năng yêu cầu và nhận kết quả từ Web Server

9

## Database Server

- Database server (máy phục vụ cơ sở dữ liệu): Máy tính mà trên đó có cài đặt phần mềm Hệ quản trị Cơ sở dữ liệu (HQTSDL).
- Một số HQTSDL: **SQL Server, MySQL, Oracle...**

10

## Web Browser

- Phần mềm dùng để xem các tài liệu hoặc tìm kiếm các tài nguyên trên Internet.
- Một số trình duyệt web thông dụng hiện nay:
  - Internet Explorer
  - Mozilla Firefox
  - Chrome
  - Opera
  - Safari

11

## URL

- URL (Uniform Resource Locator): Đường dẫn chỉ tới một trang web cụ thể trên Internet.
- Cú pháp đầy đủ:

**scheme://<host> [:port] [<path> [?<queryString>] ]**

Trong đó:

- **scheme**: tên giao thức (http, https, ftp, mailto,...)
- **host**: tên miền
- **port**: cổng dịch vụ trên máy chủ
- **path**: đường dẫn và tên của tập tin tài nguyên trên máy chủ
- **queryString**: các truy vấn, các mục con

12

## URL

- Ví dụ:

<http://www.legend.net.uk/resources/gloss.html>

<http://www.tuoitre.com.vn/Tianyon/Index.aspx?ArticleID=238657&ChannelID=3>

[http://www.htmlib.com:80/example\\_files/file.html?Lib=3](http://www.htmlib.com:80/example_files/file.html?Lib=3)

13

## HTTP

- HTTP (HyperText Transfer Protocol): Là giao thức chuyển giao siêu văn bản trên web.
- Giao thức này là tập hợp các qui định dùng để trao đổi các tài liệu (văn bản, hình ảnh, âm thanh, video, các tập tin đa truyền thông,...) giữa Web server và trình duyệt Web.

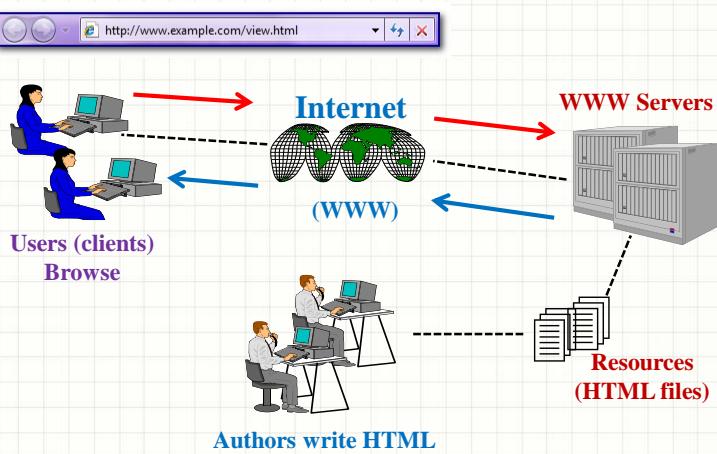
14

## Cơ chế tương tác từ người dùng đến Web Server

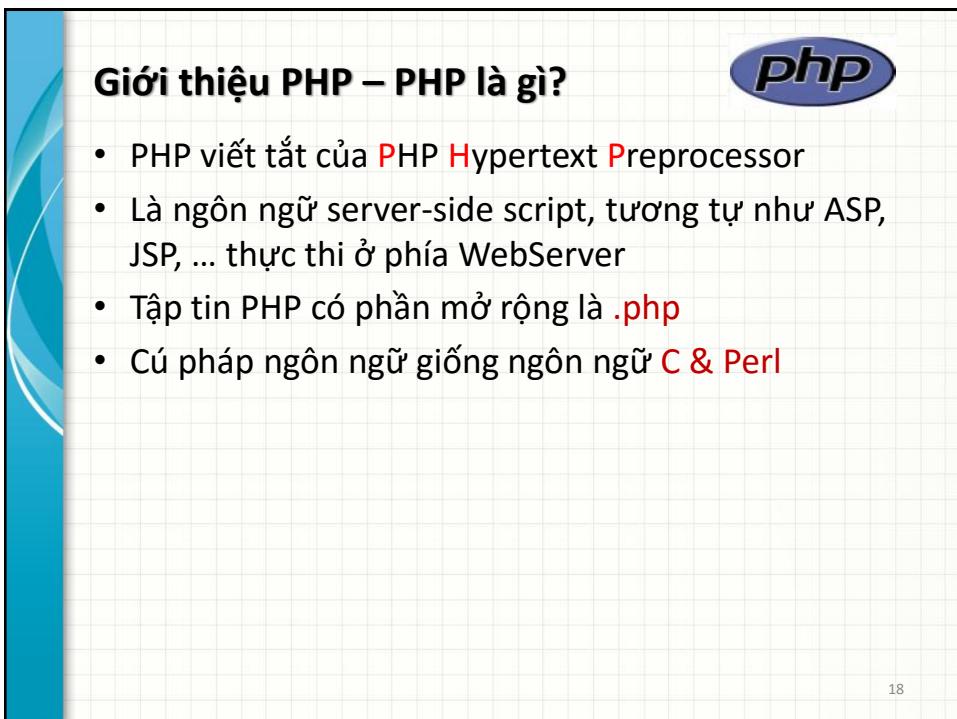
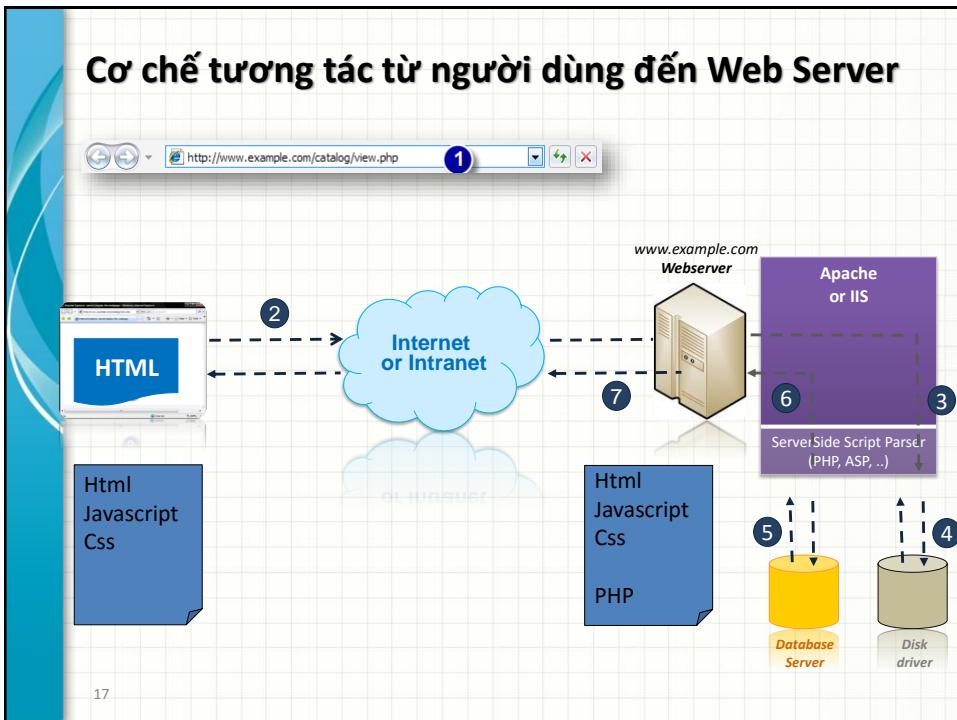


15

## Cơ chế tương tác từ người dùng đến Web Server



16



## Giới thiệu PHP - Lịch sử phát triển



- **PHP/FI:** do Rasmus Lerdorf tạo ra vào năm 1995
- **PHP/FI 2.0 (1997):** Chuyển sang ngôn ngữ script xử lý trên server. Hỗ trợ CSDL, Upload File, khai báo biến, mảng, hàm đệ quy, câu điều kiện, biểu thức,...
- **PHP 3.0 (1998):** bộ phân tích mã PHP (parser) của Zeev Suraski và Andi Gutmans, hỗ trợ ODBC, đa hệ điều hành, giao thức email (SNMP, IMAP).
- **PHP 4.0 (2000):** Hỗ trợ nhiều máy chủ Web hơn, hỗ trợ phiên làm việc HTTP. Parse đổi tên thành Zend Engine. Bổ sung các tính năng bảo mật cho PHP.
- **PHP 5.0 (2005):** Bổ sung Zend Engine II hỗ trợ lập trình HDT, XML, SOAP cho Web Services, SQLite
- Phiên bản mới nhất của PHP là version PHP 5.6.3 ([www.php.net](http://www.php.net), 13/11/2014)

19

## Giới thiệu PHP – Đặc điểm của PHP

- Miễn phí, download dễ dàng từ Internet.
- Ngôn ngữ đơn giản.
- Tốc độ xử lý nhanh, dễ sử dụng.
- Luôn được cải tiến và cập nhật (mã nguồn mở).
- Có nhiều hướng dẫn sử dụng trên mạng.
- PHP có thể thực thi trên bất cứ hệ điều hành nào.
- PHP không chỉ làm việc với HTML mà còn có thể làm việc được với hình ảnh, PDF, Flash movie,...
- Có thể dễ dàng nối kết với các HQTCSDL như: MySQL, Microsoft SQL Server 2000, Oracle, PostgreSQL, Adabas, dBase, Informix....

20

## Giới thiệu PHP – Một số website lớn

The Internet's most trafficked site

PHP at Yahoo!  
<http://www.yahoo.com>

Joomla! because open source matters  
Portal

mambo power is simplicity  
Portal

moodle Course Management System  
Course Management System

MediaWiki  
Wiki

SUGARCRM COMMERCIAL OPEN SOURCE Customer Relationship Management  
Customer Relationship Management

zen cart the art of e-commerce e-Commerce

XOOPS powered by you Portal

phpBB creating communities Bulletin Board

Dragonfly Content Management System

MANTIS Help Desk

21

## Giới thiệu PHP – Cần gì để chạy PHP?

- Download PHP  
<http://www.php.net/downloads.php>
- Download MySQL Database  
<http://www.mysql.com/downloads/index.html>
- Download Apache Server  
<http://httpd.apache.org/download.cgi>
- Download WAMP, XAMPP, LAMP
  - Download WAMP for free here:  
<http://www.wampserver.com/en/download.php>
  - Download XAMPP for free here:  
<https://www.apachefriends.org/download.html>
  - Download LAMP for free here:  
<http://sourceforge.net/projects/lampas/>

22

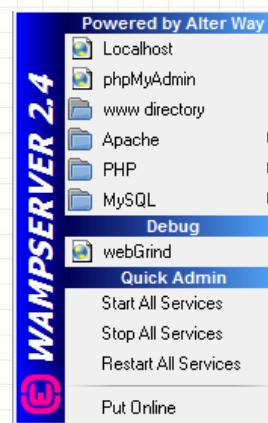
## WAMP Server

- Đặc điểm
- Cài đặt

23

## WAMP Server – Đặc điểm

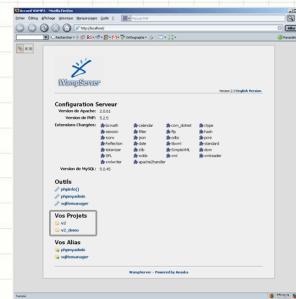
- **WAMP**: là một gói phần mềm Web Server tất cả trong một (All-in-One), gồm có: Apache, MySQL, PHP chạy trên nền Windows
  - Có thể cài đặt dễ dàng.
  - Được cập nhật đều đặn.
  - Rất thuận lợi cho việc tạo máy chủ Web để chạy thử, thiết kế Website bằng PHP.
  - Chạy ổn định cho Joomla 1.0.x, Joomla 1.5.
  - Hỗ trợ PHP5.x
  - Miễn phí



24

## WAMP Server – Cài đặt

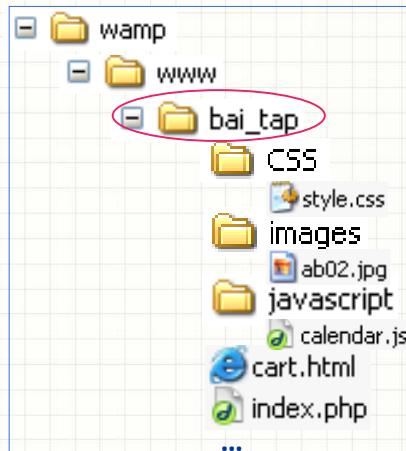
- Sử dụng phiên bản Wamp 2.0i để cài đặt lần lượt theo các bước.
- Khi cài đặt xong, chúng ta sẽ thấy biểu tượng nửa hình tròn màu trắng ở góc phải dưới của màn hình.
- Kiểm tra kết quả cài đặt
- **Chú ý: Khi cài đặt và sử dụng Wamp thì cần phải tắt IIS**



25

## Cách tổ chức và lưu trữ ứng dụng

- Thư mục lưu trữ ứng dụng được đặt trong thư mục wamp/www



26

## Cách tổ chức và lưu trữ ứng dụng

- Các loại tập tin thường gặp trong ứng dụng PHP

Tập tin	Diễn giải
.php	Tập tin mã nguồn viết theo ngôn ngữ PHP
.js	Tập tin mã nguồn viết theo ngôn ngữ JavaScript
.inc	Tập tin dùng chung
.css	Tập tin html dùng để định style cho trang web
.htm, .html	Tập tin viết bằng ngôn ngữ HTML
.jpg, .gif, ...	Các tập tin hình ảnh
.txt	Tập tin dữ liệu text
.sql	Tập tin dữ liệu sql

27

## Công cụ để xây dựng ứng dụng PHP

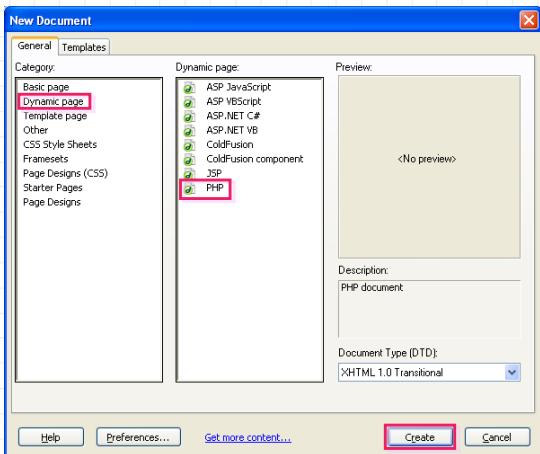
- Macro Media Dream Weaver



28

## Các thao tác cơ bản

- Tạo trang PHP



29

## Các thao tác cơ bản

- Thiết lập thuộc tính cho trang

```

<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>Trang đầu tiên</title>
</head>
<body>
</body>
</html>

```

30

## Các thao tác cơ bản

- Nhúng PHP vào HTML

A screenshot of a code editor window titled "trang\_dau\_tien.php". The code is as follows:

```
1 <head>
2 <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
3 <title>Trang đầu tiên</title>
4 </head>
5
6 <body>
7 <%php
8 echo "Đây là ngôn ngữ lập trình PHP";
9 %>
10 </body>
11 </html>
12
```

The line of code "echo "Đây là ngôn ngữ lập trình PHP";" is highlighted with a red border.

31

## BÀI 2

# PHP CƠ BẢN

- Quy ước trong PHP
- Khai báo biến, hằng
- Kiểu dữ liệu
- Toán tử
- Cấu trúc điều khiển

## Quy ước – Cú pháp

- PHP cũng có thẻ bắt đầu và kết thúc giống với ngôn ngữ HTML.

- **Cách 1 :** Cú pháp chính:

```
<?php  
    // code php  
?>
```

- **Cách 2:** Cú pháp ngắn gọn

```
<?  
    // code php  
?>
```

33

## Quy ước – Cú pháp

- **Cách 3:** Cú pháp giống với ASP.

```
<%  
    //code php  
%>
```

- **Cách 4:** Cú pháp bắt đầu bằng script

```
<script language="php">  
.....  
</script>
```

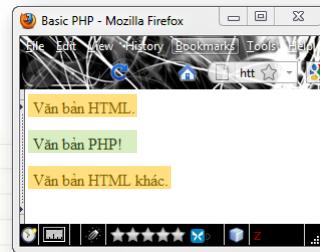
34

## Quy ước – Ví dụ 1

```

1  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
2  <html>
3      <head>
4          <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
5          <title>Basic PHP</title>
6      </head>
7      <body>
8          <p>Văn bản HTML.</p>
9
10         <?php
11             echo '<p>Văn bản PHP!</p>';
12         ?>
13         <p>Văn bản HTML khác.</p>
14     </body>
15 </html>

```



35

## Quy ước – Ví dụ 2

```

1  <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
2  <html>
3      <head>
4          <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
5          <title></title>
6      </head>
7      <body>
8          <?php
9              echo '<p>Khối dữ liệu PHP 1.</p>';
10         ?>
11         <p>Dữ liệu HTML, <?php echo 'Dữ liệu PHP 2'; ?></p>
12
13         <?php echo '<br>'; ?>
14             Một ví dụ kết hợp HTML và PHP.
15             <?php echo '<br>'; ?>
16         </body>
17 </html>

```



36

## Quy ước – Ví dụ 3

```

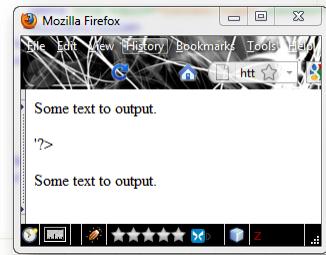
1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
2 <html>
3   <head>
4     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
5     <title></title>
6   </head>
7   <body>
8     <?=<p>Some text to output.</p>'?>
9
10    <?php
11      echo '<p>Some text to output.</p>';
12    ?>
13   </body>
14 </html>

```

→ Không nên sử dụng thẻ viết tắt

### Chỉnh sửa php.ini

Tham số	Ý nghĩa
short_open_tag = Off	Cho phép sử dụng <? ?>
asp_tags = Off	Cho phép sử dụng <% %>
post_max_size = 8M	Kích thước tối đa của dữ liệu gửi lên server
file_uploads = On	Cho phép upload file
upload_max_filesize = 2M	Kích thước tối đa của mỗi file upload

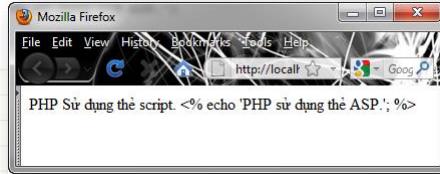


## Quy ước – Ví dụ 4

```

1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
2 <html>
3   <head>
4     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
5     <title></title>
6   </head>
7   <body>
8     <script language="php">
9       echo 'PHP Sử dụng thẻ script. ';
10      </script>
11
12      <%
13        echo 'PHP sử dụng thẻ ASP.';
14      %>
15   </body>
16 </html>

```



## Quy ước - Lệnh và ghi chú

- Mỗi lệnh phải kết thúc bằng dấu ; (trừ lệnh cuối cùng trước khóa ?>)
- Không phân biệt** khoảng trắng, Tab, xuống dòng trong câu lệnh

```
<?php print "Hello"; print " World!"; ?>
```

```
<?php  
Print "Hello" ;  
print " World!";  
?>
```

- Khối (nhiều) lệnh được đặt trong cặp { }

39

## Quy ước - Lệnh và ghi chú

- Ghi chú: Theo cú pháp của C++ và Perl
  - // chú thích có giá trị đến cuối dòng
  - # chú thích có giá trị đến cuối dòng
  - /\*

chú thích trên nhiều dòng

\*/

## Quy ước – Ví dụ ghi chú

```

1 <!--
2 To change this template, choose Tools | Templates
3 and open the template in the editor.
4 -->
5 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
6 <html>
7   <head>
8     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
9     <title></title>
10    </head>
11    <body>
12      <?php
13        # Title: My PHP program
14        # Author: Jason
15        echo "This is a PHP program";
16      ?>
17
18      <?php
19        /*
20          Title: My PHP Program
21          Author: Jason
22          Date: July 26, 2007
23        */
24      ?>
25    </body>
26  </html>
41</>
```

## Quy ước - Xuất dữ liệu ra trình duyệt

- Để xuất dữ liệu ra trình duyệt chúng ta có những dòng cú pháp sau:
- echo print printf sprintf
- Thông tin bao gồm: biến, chuỗi, hoặc lệnh HTML ....

```
<?php
$nl="PHP";
$n2="ASP.NET";
echo $n1 , "và", $n2, "là ngôn ngữ WebServer";
print ("Welcome to $n1<br>");
?>
```

```
<?php
$number = 9;
$str = "Beijing";
$txt = sprintf("There are %u million bicycles in %s.", $number, $str);
echo $txt;
?>
```

## Quy ước - Xuất dữ liệu ra trình duyệt

- Một số kiểu định dạng

Ký hiệu	Kiểu tham số	Thể hiện ra
%b	Số nguyên	Số nhị phân
%c	Số nguyên	1 ký tự ACSII
%d	Số nguyên	Số nguyên
%f	Số thực	Số thực
%o	Số nguyên	Số bát phân
%s	Chuỗi	Chuỗi
%u	Số nguyên	Số nguyên không dấu
%x	Số nguyên	Số thập lục phân thường
%X	Số nguyên	Số thập lục phân hoa

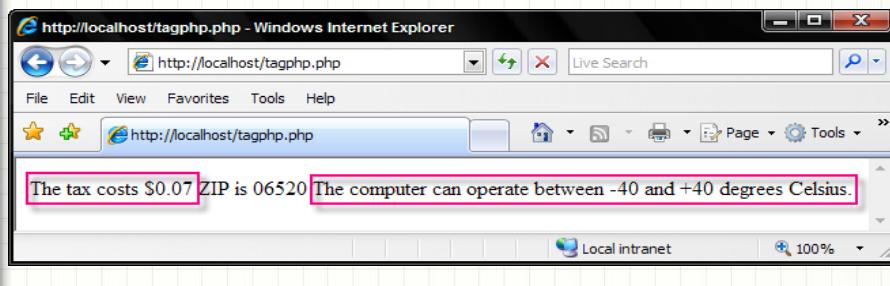
43

## Ví dụ

```
<?php
$tax = 0.075;
printf('The tax costs %.2f', $tax);

$zip = '6520';
printf("ZIP is %05d", $zip);

$min = -40; $max = 40;
printf("The computer can operate between %d and %d
degrees Celsius.", $min, $max);
?>
```



## Biến

- Khai báo biến:  
\$tên\_var = value;
- Không khai báo kiểu dữ liệu
- Biến tự khởi tạo khi được gán giá trị lần đầu
- Tên biến:
  - Chỉ bao gồm các ký tự (A..Z, a..z), ký số (0..9), \_
  - Không được bắt đầu bằng ký số (0..9)
  - Phân biệt chữ hoa – chữ thường
- Ví dụ:
  - \$size \$my\_drink\_size \$drink4you
  - \$\$2hot4u \$drink-size \$1One Size

45

## Biến

- Ví dụ:

```

1 <?
2 $a= 100 // biến a ở đây có giá trị là 100.
3 $a= "PHP is easy" // Biến a ở đây có giá trị "PHP Is easy".
4 Biena=123 //Có lỗi vì bắt đầu 1 biến phải có dấu "$"
5 $123a="PHP" //Có lỗi vì phần tên bắt đầu của biến là dạng số.
6 ?>

```

- Variable variables:
  - Cho phép thay đổi tên biến
  - Ví dụ:

\$varname = "Bien\_moi";  
 \$\$varname = "xyz"; // \$Bien\_moi = "xyz"

46

## Phạm vi hoạt động của biến

Loại	Vị trí khai báo	Phạm vi ảnh hưởng
Biến cục bộ	Trong hàm	Trong hàm
Tham số của hàm	Khai báo hàm	Trong hàm
Biến toàn cục	Sau từ khóa <b>GLOBAL</b>	Webpage
Biến tĩnh	Sau từ khóa <b>STATIC</b>	Webpage
Biến hệ thống	<b><code>\$_SERVER, \$_POST, ...</code></b>	Website

47

## Phạm vi hoạt động của biến

- Biến cục bộ
  - Biến được khai báo trong hàm
  - Khi ra khỏi hàm => biến cục bộ và giá trị của nó sẽ bị hủy bỏ

```
<?php
$a = 1; // phạm vi toàn cục
function Test()
{
    echo $a; // phạm vi cục bộ
}
Test(); → không có
echo $a; → 1
?>
```

48

## Phạm vi hoạt động của biến

- Biến toàn cục
  - Có thể truy xuất bất cứ nơi nào trong trang
  - Khi muốn sử dụng và cập nhật biến toàn cục trong hàm thì phải dùng từ khóa **global** phía trước biến hoặc dùng **`$_GLOBALS["tên biến"]`**

```
<?php
$a = 1;
$b = 2;
function Sum()
{
    $_GLOBALS['b'] = $_GLOBALS['a'] + $_GLOBALS['b'];
}
Sum();
echo $b; → 3
?>
```

```
<?php
global $a, $b;
$b = $a + $b;
}
Sum();
echo $b; → 3
?>
```

## Phạm vi hoạt động của biến

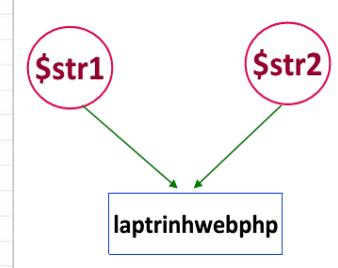
- Biến static
  - Không mất đi giá trị khi ra khỏi hàm
  - Sẽ giữ nguyên giá trị trước đó khi hàm được gọi một lần nữa
  - Phía trước tên biến static phải có từ khóa **static**

```
<?php
function Test()
{
    static $a = 0;
    echo $a;
    $a++;
}
Test(); → 0
Test(); → 1
Test(); → 2
?>
```

## Tham chiếu

- Tham chiếu trong PHP cho phép tạo ra hai hay nhiều biến có cùng một nội dung.
- Ví dụ:

```
<?php
$str1 = "laptrinhwebphp";
$str2 = &$str1;
echo $str1; → laptrinhwebphp
echo $str2; → laptrinhwebphp
?>
```



## Hằng

- Định nghĩa:
  - Cú pháp: **define**('tên\_hằng', giá trị);
  - Giá trị hằng chỉ được dùng các kiểu dữ liệu cơ bản (boolean, integer, float, string)
  - Bắt buộc định nghĩa trước khi dùng
- Quy ước về cách đặt tên:
  - **Giống** cách đặt tên biến
  - Không sử dụng ký hiệu **\$**
  - Thường đặt tên bằng chữ **in hoa**

## Hàng

- Ví dụ:

```
<? Php
define ("C", "COMPANY");
define ("A", 3.1415);
echo "Giá trị của C là:" . C;
?>
```

53

## Kiểu dữ liệu trong PHP

- 1 Biến trong PHP có thể lưu bất kỳ kiểu dữ liệu nào
- Trong PHP có các kiểu dữ liệu chính sau:

Kiểu Dữ Liệu	Ví dụ	Mô Tả
Integer	10	Một số nguyên
Double	5.208	Kiểu số thực
String	"How are you ?"	Một tập hợp các ký tự
Boolean	True or False	Giá trị true hoặc false
Object	Hướng đối tượng trong PHP	
Array	Mảng trong PHP, chứa các phần tử.	

54

## Kiểu dữ liệu trong PHP

- Chuyển kiểu dữ liệu:

- Cách 1: (**automatic**)

`$var = "100" + 15;`

`$var = "100" + 15.0;`

`$var = 39 + " Steps";`

- Cách 2: (**datatype**) \$var

- Cách 3: **settype(\$var, “datatype”)**

\$var	(int)\$var	(bool)\$var	(string)\$var
<b>null</b>	0	false	“”
<b>true</b>	1		“1”
<b>false</b>	0		“”
“6 feet”	6	true	
“foo”	0	true	

## Các hàm kiểm tra kiểu dữ liệu của biến

- Kiểm tra tồn tại **isset()**
- Kiểm tra giá trị rỗng **empty()**
- Kiểm tra trị kiểu số **is\_numeric()**
- Kiểm tra kiểu dữ liệu của biến **is\_int()**, **is\_long()**, **is\_double()**, **is\_string()**
- Xác định kiểu của biến **gettype()**

## Kiểm tra tồn tại: isset()

- Ý nghĩa: dùng để kiểm tra biến có giá trị hay không
- Cú pháp: **isset(<tên biến 1>, <tên biến 2>, ...)**
- Kết quả trả về:
  - TRUE: nếu tất cả các biến đều có giá trị
  - FALSE: nếu một biến bất kỳ không có giá trị

```
<?php
    if(isset($_POST["bt_xac_nhan"], $_POST["ten_dn"]))
        echo "Xin chào ".$_POST["ten_dn"];
    else
        echo "Vui lòng nhập tên đăng nhập";
?>
```

57

## Kiểm tra giá trị rỗng: empty()

- Ý nghĩa: dùng để kiểm tra biến có giá trị rỗng hay không
- Cú pháp: **empty(<tên biến>)**
- Kết quả trả về:
  - TRUE: nếu biến có giá trị rỗng
  - FALSE: nếu biến có giá trị khác rỗng
- Các giá trị được xem là rỗng:
  - “ ”(chuỗi rỗng), **NULL**
  - **0** (khi kiểu là integer), **FALSE**, **array()**
  - **var \$var** (biến trong lớp được khai báo nhưng không có giá trị)

58

## Kiểm tra trị kiểu số: is\_numeric()

- Ý nghĩa: dùng để kiểm tra biến có giá trị kiểu số hay không
- Cú pháp: **is\_numeric(<tên\_var>)**
- Kết quả trả về:
  - TRUE: nếu biến có giá trị kiểu số
  - FALSE: nếu biến có giá trị không phải kiểu số
- Một số hàm kiểm tra kiểu dữ liệu của biến:
  - **is\_int(), is\_long()**
  - **is\_string(), is\_double()**
  - Cú pháp chung: **tên\_hàm(<tên\_var>)**

59

## Xác định kiểu của biến: gettype()

- Ý nghĩa: kiểm tra biến hoặc giá trị có kiểu dữ liệu nào: integer, string, double, array, object, class, ...
- Cú pháp: **gettype(<tên\_var>)** hoặc **<giá\_trị>**
- Kết quả trả về:
  - Kiểu của giá trị hay kiểu của biến

```

1 <?php
2 $a= 5;
3 echo gettype($a); // Integer.
4 $a="ghonline.info";
5 echo gettype($a); //String
6 ?>
7 |

```

60

## Kiểu số - int, float

- Một số hàm xử lý số
 

– abs	pow	decbin	srand(seed)
– ceil	sqrt	bindec	rand
– floor	log	dechex	rand(min, max)
– round	log10	hexdec ...	
- Ví dụ:
  - Phát sinh một “mầm” ngẫu nhiên  
\$seed = (float) microtime( ) \* 100000000;
  - Khởi tạo bộ phát sinh số ngẫu nhiên  
srand(\$seed);
  - In con số ngẫu nhiên  
print rand(); // Giá trị ngẫu nhiên từ 0 đến getrandmax()  
print rand(1, 6); // Giá trị ngẫu nhiên từ 1 đến 6

61

## Kiểu chuỗi

- Ví dụ:
    - 'Hello ', "welcome to VietNam"
  - Để tạo 1 biến chuỗi, chúng ta phải gán giá trị chuỗi cho 1 biến hợp lệ.
- Ví dụ:
- \$fisrt\_name= "Nguyen";
  - \$last\_name= 'Van A';
- Toán tử nối chuỗi: dấu ". "
  - Ví dụ:
 

```
<?php
$name="X";
echo "Hello " . $name;
?>
```

62

## Kiểu chuỗi

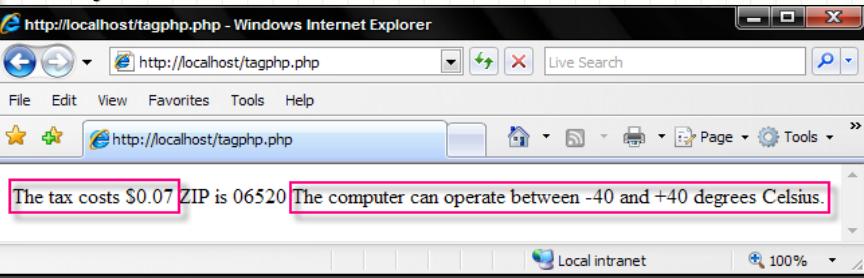
- Một số hàm xử lý chuỗi:
  - printf
  - str\_pad
  - strlen
  - ...

trim  
str\_replace  
substr

strtolower  
strtoupper  
strcasecmp

63

## Ví dụ



```
<?
$tax = 0.075;
printf('The tax costs %.2f', $tax);

$zip = '6520';
printf("ZIP is %05d", $zip);

$min = -40; $max = 40;
printf("The computer can operate between %d and %d
degrees Celsius.", $min, $max);
?>
```

**Ví dụ**

```

<?
echo "<pre>"

// Print a heading
echo str_pad("Dramatis Personae", 50, " ", STR_PAD_BOTH) . "\n";

// Print an index line
echo    str_pad("DUNCAN, king of Scotland", 30, ".")
        . str_pad("Larry", 20, ".", STR_PAD_LEFT)
        . "\n";
echo "</pre>"
?>

```

**Kiểu mảng - array**

- Numbered array

```

$words = array("Web", "Database", "Applications");
echo $words[0]; //Web

```

```

$numbers = array (1=>"one", "two", "three", "four");
echo $numbers[1]; //one

```

- Associated array

```

$array = array("first"=>1, "second"=>2, "third"=>3);
echo $array["second"]; //2

```

## Kiểu mảng - array

- Một số hàm xử lý trên mảng
    - count in\_array sort asort ksort usort
    - min array\_reverse rsort arsort krsprt uasort
    - max uksort
  - Ví dụ:
- ```
$dinner = array( 'Sweet Corn and Asparagus', 'Lemon Chicken',
  'Braised Bamboo Fungus');

sort($dinner);
print "I want $dinner[0] and $dinner[1].";
$dishes = count($dinner);
print $dishes;
//I want Braised Bamboo Fungus and Lemon Chicken. 3
```

67

## Kiểu mảng - array

- Một số hàm liên quan đến mảng
  - **reset(array)**
  - **array\_push(array, elements)** : Thêm elements vào cuối mảng
  - **array\_pop(array)** : Lấy phần tử cuối ra khỏi mảng
  - **array\_unshift(array, elements)** : Thêm elements vào đầu mảng
  - **array\_shift(array)** : Lấy phần tử đầu ra khỏi mảng
  - **array\_merge(array, array)** : kết 2 mảng lại và trả ra mảng mới
  - **shuffle(array)** : Sort random mảng
  - **sort(array, flag):** flag = {sort\_regular, sort\_numeric, sort\_string, sort\_locale\_string}

68

## Các phép toán trong PHP

- Toán tử gán và số học

| Gán | Số học | Kết hợp |
|-----|--------|---------|
| =   | +      | + =     |
|     | -      | - =     |
|     | *      | * =     |
|     | /      | / =     |
|     | %      | % =     |

69

## Toán tử so sánh

| Ký hiệu | Ý nghĩa                     |
|---------|-----------------------------|
| ==      | Bằng giá trị                |
| == =    | Bằng giá trị và cùng kiểu   |
| !=      | Khác giá trị                |
| <>      | Khác giá trị                |
| ! ==    | Khác giá trị hoặc khác kiểu |
| <       | Nhỏ hơn                     |
| >       | Lớn hơn                     |
| <=      | Nhỏ hơn hoặc bằng           |
| >=      | Lớn hơn hoặc bằng           |

70

## Toán tử logic

| Ký hiệu | Ý nghĩa |
|---------|---------|
| and     | And     |
| &&      | And     |
| or      | Or      |
|         | Or      |
| xor     | Xor     |
| !       | Not     |

71

## Toán tử xử lý bit

| Ký hiệu | Ý nghĩa   |
|---------|-----------|
| &       | And       |
|         | Or        |
| ^       | Xor       |
| ~       | Not       |
| <<      | Dịch trái |
| >>      | Dịch phải |

72

## Toán tử tăng giảm 1

| Ký hiệu | Ý nghĩa |
|---------|---------|
| ++      | Tăng 1  |
| --      | Giảm 1  |

73

## Các cấu trúc điều khiển

- Cấu trúc rẽ nhánh (if)
- Cấu trúc chọn lựa (switch)
- Cấu trúc lặp (for, foreach, while, do while)
- Sử dụng break và continue trong cấu trúc lặp

74

## Câu lệnh *if*

- Cú pháp:

```
if (biểu thức điều kiện)
```

```
    khối lệnh 1;
```

```
else
```

```
    khối lệnh 2;
```

75

## Ví dụ

```
1 <?php
2 $a=5;
3 $b=7;
4 if($a < $b)
5 {
6     echo "Bien A co gia tri nho hon bien B";
7 }
8 else
9 {
10    echo "Bien A co gia tri lon hon bien B";
11 }
12 ?>
```

76

## Câu lệnh if

- Các câu lệnh if có thể lồng nhau

```

if (điều kiện 1)
{
    khối lệnh1
}
elseif (điều kiện2)
{
    khối lệnh2
}
...
else
{
    khối lệnh khi không thỏa các điều kiện trên
}

```

77

## Toán tử ?:

- Cú pháp:

(điều kiện) ? <kết quả khi điều kiện đúng> : <kết quả khi điều kiện sai>

- Ý nghĩa: dùng để thay thế cho cấu trúc điều khiển if...else với một câu lệnh bên trong
- Có thể lồng nhiều toán tử ?: với nhau

&lt;?php

```

$a = $_POST["a"];
$b = $_POST["b"];
if($a>$b)
    $so_lon = $a;
else
    $so_lon = $b;

```

?&gt;

&lt;?php

```

$a = $_POST["a"];
$b = $_POST["b"];
$so_lon = ($a>$b)?$a:$b;
?
```

78

## Câu lệnh chọn lựa **switch**

```
switch (biểu thức)
{
    case biểu thức 1:
        khối lệnh 1;
        break;
    case biểu thức 2:
        khối lệnh 2;
        break;
    ...
    case biểu thức n:
        khối lệnh n;
        break;
    default:
        khối lệnh cuối;
}
```

79

## Vòng lặp **while/do...while**

- **while** (biểu thức điều kiện)  
khối lệnh;
- **do**  
khối lệnh;  
**while** (biểu thức điều kiện);

80

## Ví dụ **while/do...while**

```
<?php
    $i = 1;
    while ( $i <= 10 )
    {
        echo $i, "\n";
        $i++;
    } //end while
?>
```

```
<?php
    $i = 1;
    do
    {
        echo $i, "\n";
        $i++;
    }
    while ( $i < 10 );
?>
```

81

## Vòng lặp **for**

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

khối lệnh;

- biểu thức 1: thực hiện 1 lần khi bắt đầu vòng lặp
- biểu thức 2: điều kiện lặp, được xét trước mỗi lần lặp
- biểu thức 3: thực hiện sau mỗi lần lặp

```
<?php
    for ( $i = 0; $i < 10; $i++ )
    {
        echo $i, "\n";
    } //end for
?>
```

82

## Vòng lặp **foreach**

- Công dụng:
  - Câu lệnh **foreach** được dùng để duyệt tập hợp (mảng)
- Cú pháp duyệt giá trị các phần tử trong mảng:  
`foreach ( $ten_mang as $value )`  
 câu lệnh;
- Cú pháp duyệt cả khóa và giá trị các phần tử trong mảng:  
`foreach ( $ten_mang as $key => $value )`  
 câu lệnh;

83

## Ví dụ

- `<?php`

```
$a = array('a' => 1, 'b' => '2', 'c' => '3');
foreach ( $a as $value )
    echo $value, "<br>";
```

`?>`

- `<?php`

```
$a = array('a' => 1, 'b' => '2', 'c' => '3');
foreach ( $a as $key => $value )
{
    echo "a[$key] = $value <br>";
} //end foreach
```

`?>`

84

## Sử dụng break và continue trong cấu trúc lặp

- **break:** thoát khỏi cấu trúc điều khiển dựa trên kết quả của biểu thức luận lý kèm theo (điều kiện kiểm tra)

```
<?php
// kiểm tra số nguyên tố
$so = $_POST["so"];
$kq = true;
for($i=2; $i<$so; $i++)
{
    if($so%$i==0)
    {
        $kq = false;
        break;
    }
}
?>
```

85

## Sử dụng break và continue trong cấu trúc lặp

- **continue:** Khi gặp continue, các lệnh bên dưới continue tạm thời không thực hiện tiếp, khi đó con trỏ sẽ nhảy về đầu vòng lặp để kiểm tra giá trị của biểu thức điều kiện còn đúng hay không

```
<?php // tính tổng các số lẻ từ 1 đến 10
$tong =0;
for($i=1;$i<=10;$i++)
{
    if($i%2==0)
        continue;
    $tong = $tong + $i;
}
echo $tong; → 25
?>
```

86

## BÀI 3

# FORM VÀ CƠ CHẾ TRUYỀN NHẬN DỮ LIỆU

- Form và Form field
- Cơ chế truyền nhận dữ liệu

## Giới thiệu Form

- Được dùng để nhận dữ liệu từ phía người dùng
- Giúp gởi yêu cầu của người dùng đến trang xử lý trong ứng dụng web
- Tag `<form>` dùng để chứa các thành phần khác của form
- Những thành phần nhập liệu được gọi là Form Field
  - text field
  - password field
  - multiple-line text field
  - .....

## Ví dụ

The screenshot shows a Yahoo! sign-up page in Internet Explorer. The form is titled "Chào Bạn!" and asks for basic personal information. It includes fields for Name (Tên), Gender (Giới Tính), Birthdate (Ngày Tháng Năm Sinh), Location (Tôi sống tại), and Password (Mật Khẩu). There are also fields for a secondary email (Email Thay Thế Khác) and a security question (Câu Hỏi Bảo Mật). A note at the bottom explains how to recover an account if forgotten.

## Tag <Form>

- Là container chứa các thành phần nhập liệu khác.
- Cú pháp:

```
<form name="tên_form" action="đường dẫn đến file script"
      method="cách gửi dữ liệu đến script"
      <!– Các thành phần của form -->

</form>
```

- Name:** tên form
- Action:** chỉ định trang web nhận xử lý dữ liệu từ FORM này khi có sự kiện click của button **SUBMIT**.
  - Đường dẫn đến file script phải là đường dẫn tương đối
  - Trường hợp script nằm cùng file với FORM thì khai báo **action=<?php echo \$\_SERVER['PHP\_SELF'] ?>**
- Method:** Xác định phương thức chuyển dữ liệu (**POST, GET**)

## Tag <Form> - Ví dụ

Dangnhap.htm

```
<html>
  <body>
    <form Name="Dangnhap"
          Action="/admin/xlDangnhap.php"
          Method="Post">
      .....
    </form>
  </body>
</html>
```

## Các thành phần của Form

- Gồm các loại FormField sau:
  - Text field
  - Password field
  - Hidden Text field
  - Check box
  - Radio button
  - File Form Control
  - Submit Button, Reset Button, Generalized Button
  - Multiple-line text field
  - Label
  - Pull-down menu
  - Scrolled list
  - Field Set

## Thẻ <input>

```
<INPUT
    TYPE = "KiểuLoại"
    NAME = string
    VALUE = string
    .....
    >
```

- Type:
  - text / password/ radio/ checkbox
  - hidden
  - submit / button

First name:   
 Last name:

Password:

Male  
 Female

I have a bike  
 I have a car

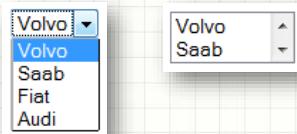
Username:

93

## Thẻ <select>

```
<SELECT name="???" multiple>
<optgroup label="string">
    <option value=""> XYZ </option>
    <option value="" selected> 123 </option>
</optgroup>
.....
</SELECT>
```

- Type:
  - Combobox
  - Listbox

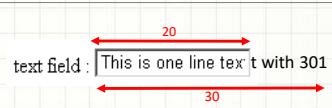


94

## Các thành phần của Form – Text Field

- Dùng để nhập một dòng văn bản
- Cú pháp

```
<INPUT
    TYPE      = "TEXT"
    NAME     = string
    READONLY
    SIZE      = variant
    MAXLENGTH = long
    TABINDEX
    VALUE     = string
    .....
    >
```



- Ví dụ

```
<input type="text" name="txtName" value="This is one line
text with 301" size="20" maxlength="30">
```

## Các thành phần của Form – Password Field

- Dùng để nhập mật khẩu
- Cú pháp

```
<INPUT
    TYPE      = "PASSWORD"
    NAME     = string
    READONLY
    SIZE      = variant
    MAXLENGTH = long
    TABINDEX
    VALUE     = string
    .....
    >
```

password field : 

- Ví dụ

```
<input type="Password" name="txtPassword"
value="123456abc1234" size="20" maxlength="30">
```

## Các thành phần của Form – Check box

- Cú pháp

```
<input  
    TYPE      = "checkbox"  
    NAME     = "text"  
    VALUE    = "text"  
    [checked]  
>
```

- Ví dụ

```
<html>  
    <body>  
        Check box group : <br>  
        Anh van: <input type="checkbox" name="Languages1"  
value="En"><br>  
        Hoa: <input type="checkbox" name="Languages2" value="Chz"  
checked><br>  
        Nhung: <input type="checkbox" name="Languages3"  
value="Jp"><br>  
    </body>  
</html>
```



## Các thành phần của Form – Radio button

- Cú pháp

```
<input  
    TYPE      = "radio"  
    NAME     = "text"  
    VALUE    = "text"  
    [checked]  
>
```

- Ví dụ

```
<html>  
    <body>  
        Radio Button Group : <br>  
        Nam: <input type="radio" name="sex" value="nam" checked><br>  
        Nu: <input type="radio" name="sex" value="nu" checked><br>  
    </body>  
</html>
```

```
<html>  
    <body>  
        Radio Button Group : <br>  
        Nam: <input type="radio" name="sex1" value="nam" checked><br>  
        Nu: <input type="radio" name="sex2" value="nu" checked><br>  
    </body>  
</html>
```



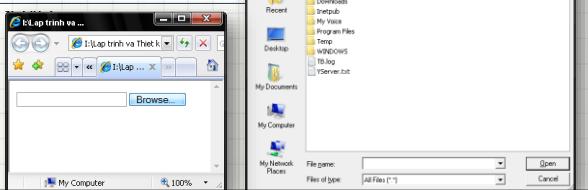
## Các thành phần của Form – File Form Control

- Dùng để upload 1 file lên server
- Cú pháp

```
<form action="..." method="post" enctype="multipart/form-data"
      name="...">
    <input TYPE="FILE" NAME="...">
</form>
```

- Ví dụ

```
<html>
<body>
  <form name="frmMain" action="POST" enctype="multipart/form-data">
    <input type="file" name="fileUpload">
  </form>
</body>
</html>
```



## Các thành phần của Form – Submit button

- Nút phát lệnh và gửi dữ liệu của form đến trang xử lý.
- Mỗi form chỉ có một nút submit và nút này được viền đậm
- Cú pháp:

```
<input TYPE="submit" name="..." value="...">
```

- Ví dụ:

```
<input type="submit" name="btnSend" value="Send">
```



## Các thành phần của Form – Reset Button

- Dùng để trả lại giá trị mặc định cho các control khác trong form
- Cú pháp  

```
<input TYPE="reset" name="..." value="...">
```
- Ví dụ

```
<input type="reset" name="btnReset" value="Rest">
```



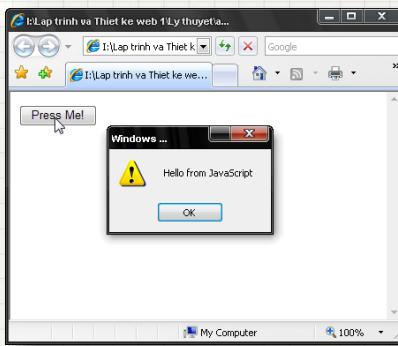
## Các thành phần của Form - Generalized Button

- Cú pháp

```
<input type="button" name="..." value="..." onclick="script">
```

- Ví dụ

```
<input type="button" name="btnNormal"
       value="Press Me!" onclick="alert('Hello from
JavaScript');">
```



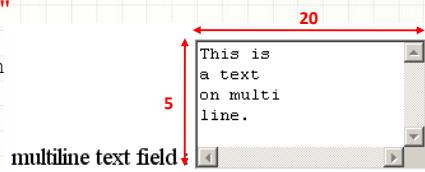
## Các thành phần của Form – Multiline Text Field

- Dùng để nhập văn bản nhiều dòng
- Cú pháp

```
<TEXTAREA
    COLS          = long
    ROWS          = long
    DISABLED
    NAME          = string
    READONLY
    TABINDEX      = integer
    WRAP          = OFF | PHYSICAL | VIRTUAL> .....
```

- Ví dụ

```
<textarea cols="20" rows="5"
    wrap="off">
    This is a text on multilin
</textarea>
```



## Các thành phần của Form - Label

- Dùng để gán nhãn cho một Form Field
- Cú pháp

```
<LABEL
    FOR = IDString
    CLASS=string
    STYLE=string
>
```

- Ví dụ

```
<label for="Languages">Anh văn: </label>
<input type="checkbox" name="Languages"
id="Languages" value="Eng">
```

Anh văn:

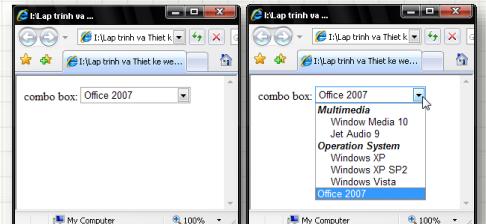
## Các thành phần của Form – Pull-down Menu

- Dùng để tạo ra một combo box
- Cú pháp

```
<Select name="...">
    <optgroup label="...">
        <option [selected] value="..." >....</option>
        .....
    </optgroup>
    <option [selected] value="..." >....</option>
    .....
</select>
```

## Các thành phần của Form – Pull-down Menu

```
<html>
    <body>
        combo box:
        <select name="DSSoftware">
            <optgroup label="Multimedia">
                <option value="WM10">Window Media 10</option>
                <option value="JA9">Jet Audio 9</option>
            </optgroup>
            <optgroup label="Operation System">
                <option value="WXP">Windows XP</option>
                <option value="WXPSP2">Windows XP SP2</option>
                <option value="WVT">Windows Vista</option>
            </optgroup>
            <option selected value="Office07">Office 2007</option>
        </select>
    </body>
</html>
```



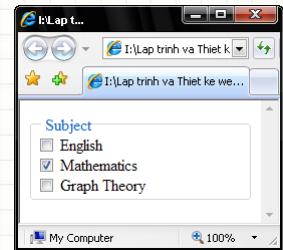
## Các thành phần của Form – Field Set

- Dùng để tạo ra Group box, nhóm các thành phần nhập liệu trong form
- Cú pháp

```
<fieldset>
    <legend>GroupBox's Name</legend>
    <input .....>
    ...
</fieldset>
```

- Ví dụ

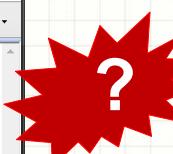
```
<html>
<body>
    <fieldset>
        <legend>Subject</legend>
        <input type="checkbox" name="Subjects" value="Eng"> English<br>
        <input type="checkbox" name="Subjects" value="Math" checked>
        Mathematics<br>
        <input type="checkbox" name="Subjects" value="GraphTheory"> Graph
        Theory<br>
    </fieldset>
</body>
</html>
```



## Cơ chế truyền nhận dữ liệu

Đăng nhập hệ thống

Tên đăng nhập :   
Mật khẩu :



Kết quả đăng nhập

Tên đăng nhập là : admin  
Mật khẩu là : phpAdmin

## Cơ chế truyền nhận dữ liệu

- Trang web **nhập** dữ liệu :
  - Sử dụng đối tượng **<form>**
  - Nhập liệu thông qua các **form field**
  - Thực hiện việc truyền dữ liệu thông qua **Submit**

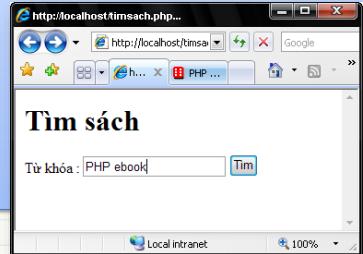
```
<FORM ACTION="URL" METHOD="GET/POST">
...
<input type="submit" value="Xử lý">
</FORM>
```

- Trang web **nhận** dữ liệu (**URL**): Sử dụng các biến toàn cục của PHP
  - \$\_POST["FieldName"]**
  - \$\_GET["FieldName"]**
  - \$\_REQUEST["FieldName"]**

## Cơ chế truyền nhận dữ liệu – Trang web nhập liệu

### timSach.php

```
<html>
  <body>
    <h1>Tìm sách</h1>
    <form action="x1TimSach.php" Method="GET" >
      Từ khóa : <input type="text" name="txtTukhoa"/>
      <input type="submit" value="Tìm"/>
    </form>
  </body>
</html>
```



## Cơ chế truyền nhận dữ liệu – Trang web nhập liệu

```
xltimSach.php
```

```
<html>
<body>
<?php
    $sTukhoa = $_REQUEST["txtTukhoa"]
?>
<h1>Tim sách</h1>
Từ khóa tìm sách là : <?php echo $sTukhoa; ?>
<br />
Kết quả tìm là :
</body>
</html>
```

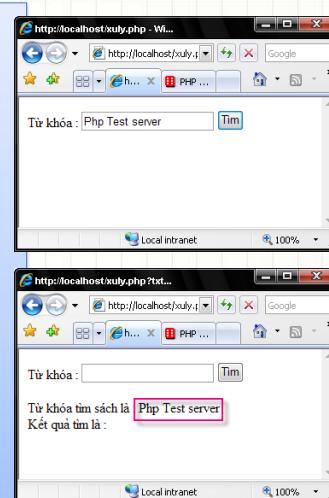


## Cơ chế truyền nhận dữ liệu – Trang web nhập liệu

```
xuly.php
```

```
<html>
<body>
<form action="xuly.php" Method="GET" >
    Từ khóa : <input type="text"
    name="txtTukhoa"/>
    <input type="submit" value="Tim"/>
</form>

<?php
    $sTukhoa = $_REQUEST["txtTukhoa"];
    if (isset($sTukhoa))
    {
        print "Từ khóa tìm sách là : ";
        $sTukhoa";
        echo "<br>Kết quả tìm là : ";
    }
?>
</body>
</html>
```



## Truyền/Nhận qua Phương thức GET

- Tham số truyền đi qua địa chỉ URL
  - <http://domain/pathfile?fieldname1=value1&fieldname2=value2>
  - Ví dụ:
    - <http://localhost/xuly.php?txtTukhoa=PHPHost>
- Nhận dữ liệu thông qua biến toàn cục của PHP
  - `$_GET["FieldName"]`
  - `$_REQUEST["FieldName"]`

## Phương thức GET – Ưu khuyết điểm

- **Khuyết điểm**
  - Không thích hợp để truyền dữ liệu có tính **bảo mật** (password)
  - **Dung lượng** Dữ liệu truyền đi có **giới hạn**
  - URL submit bằng phương thức GET được lưu lại trên server
- **Ưu điểm**
  - Người dùng có thể **bookmark** địa chỉ URL
  - Người dùng có thể **Giả lập phương thức GET** để truyền dữ liệu mà không cần thông qua FORM

## Ví dụ

**File: GET.HTM**

```
<HTML>
<HEAD>
<TITLE>Input data</TITLE>
</HEAD>

<BODY>
<IMG SRC="images/N72.jpg"><br>
<A HREF='chitiet.php?Ma=N72'>Xem chi tiết</A>
</BODY>
</HTML>
```

## Ví dụ (tt)

**File: CHITIET.PHP**

```
<HTML>
<HEAD>
<TITLE>Input data</TITLE>
</HEAD>
<BODY>
<?php
    echo "Ma dien thoai la " . $_GET["Ma"];
?>
</BODY>
</HTML>
```

## Truyền/Nhận qua Phương thức POST

- Tham số truyền đi được ẩn bên trong FORM
- Nhận dữ liệu thông qua biến toàn cục của PHP
  - `$_POST["FieldName"]`
  - `$_REQUEST["FieldName"]`

## Phương thức POST – Ưu khuyết điểm

- **Ưu điểm**
  - **Bảo mật** hơn phương thức GET
  - **Không giới hạn dung lượng** dữ liệu truyền đi
- **Khuyết điểm**
  - Kết quả trang web trả về **không thể bookmark**
  - Có thể **gây ra lỗi** nếu người dùng muốn quay lại trang kết quả (nhấn nút Back hoặc Refresh) do **bị expired**
  - Dữ liệu có thể **không truyền đi** được do vấn đề về **security**

## Chuyển dữ liệu với GET và POST

GET	POST
<p>Dữ liệu được gắn thêm vào URL khi gọi script</p> <p>Các dữ liệu được đưa vào biến mảng siêu toàn cục \$_GET với khóa tương ứng với tên các thành phần input trong form</p> <p>Nên dùng trong trường hợp dữ liệu chỉ dùng để truy vấn, không đòi hỏi bảo mật</p> <p>Không hỗ trợ uploading file</p> <p>Chỉ hỗ trợ bảng mã ASCII chuẩn</p>	<p>Dữ liệu được nhúng vào trong HTTP request khi gửi đến server</p> <p>Các dữ liệu được đưa vào biến mảng siêu toàn cục \$_POST với khóa tương ứng với tên các thành phần input trong form</p> <p>An toàn hơn so với khi dùng GET nên được dùng phổ biến hơn</p> <p>Hỗ trợ uploading file</p> <p>Hỗ trợ nhiều bảng mã</p>

119

## Bài tập áp dụng

- Tạo 1 trang web với hộp thoại nhập liệu username và password.
  - Nếu người sử dụng nhập thông tin username/password là admin/12345 thì xuất ra thông báo "welcome, admin" với kiểu chữ Tahoma, màu đỏ.
  - Ngược lại nếu nhập sai thì xuất thông báo "Username hoặc password sai. Vui lòng nhập lại".

**Đáp án:****Tạo trang login.html với nội dung sau:**

```

2 <html>
3 <head>
4 <title>login page</title>
5 </head>
6 <body>
7 <form action="checklogin.php" method=POST>
8 Username <input type=text name="user" size=15>
9 <br>Password <input type=password name="pass" size=15>
10 <br><input type=submit name=submit value="Login">
11 </form>
12 </body>
13 </html>
```

**Tiếp tục tạo trang checklogin.php với nội dung sau:**

```

2 <?php
3 $username=$_POST['user'];
4 $password=$_POST['pass'];
5 if($username == "admin" && $password=="12345")
6 {
7     echo "<font color=red>Welcome to, ".$username."<font>";
8 }
9 else
10 {
11     echo "<font color=red>Username hoac password khong chinh xac,
12 vui long dang nhap lai<font>";
13 }
14 ?>
```

## Một số thao tác xử lý file

- **Mở 1 file:** fopen("Đường dẫn", thuộc tính)
  - Đường dẫn chính là đường dẫn tới file cần mở.
  - Thuộc tính bao gồm các quyền hạn cho phép thao tác trên file đó như thế nào.
- **Các thuộc tính cơ bản :**

Mode	Điễn giải
r	Read only
r+	Read_Write
w	Write Only
w+	Write_Read. Nếu file này tồn tại, nội dung cũ sẽ bị xoá đi, còn nếu không tồn tại nó sẽ được tạo mới
a	Mở dưới dạng append dữ liệu, chỉ có write, nếu file tồn tại, sẽ ghi tiếp vào phần dưới của nội dung có sẵn, nếu file không tồn tại sẽ được tạo mới.
a+	Mở dưới dạng append dữ liệu (write và read), nếu file tồn tại, dữ liệu sẽ ghi tiếp vào phần bên dưới của nội dung cũ, nếu file không tồn tại sẽ được tạo mới
b	Mở dưới chế độ file binary

## Một số thao tác xử lý file

- **Mở 1 file**

Ví dụ:

```
<?php
$fp=fopen("test.txt",r)or exit("khong tim thay file can mo");
?>
```

- **Đóng 1 file:** fclose(file vừa mở)

Ví dụ:

```
<?php
$fp=fopen("test.txt",r)or exit("khong tim thay file can mo");
fclose($fp);
?>
```

## Một số thao tác xử lý file

- Đọc 1 file trong PHP

– Đọc file theo từng dòng: **fgets(file vừa mở)**

Ví dụ:

```
<?php
$fp=fopen("test.txt",r)or exit("khong tim thay file can mo");
echo fgets($fp);
fclose($fp);
?>
```

– Đọc file theo từng ký tự: **fgetc(file vừa mở)**

Ví dụ:

```
<?php
$fp=fopen("test.txt",r)or exit("khong tim thay file can mo");
echo fgetc($fp);
fclose($fp);
?>
```

## Một số thao tác xử lý file

- Kiểm tra đã đến cuối file chưa: **feof(file vừa mở)**

Ví dụ:

```
<?php
$fp=fopen("test.txt",r)or exit("khong tim thay file can mo");
while(!feof($fp))
{
    echo fgets($fp);
}
fclose($fp);
?>
```

## Một số thao tác xử lý file

- Ghi dữ liệu vào 1file: **fwrite("file vừa mở", "Nội dung cần ghi vào file")**

Ví dụ:

```
<?php  
$fp=fopen("test.txt",a)or exit("khong tim thay file can mo");  
$news="Trung Tam Tin hoc Ngoai ngu";  
fwrite($fp,$news);  
fclose($fp);  
?>
```

## BÀI 4

# MẢNG, CHUỖI & HÀM TRONG PHP

## Nội dung

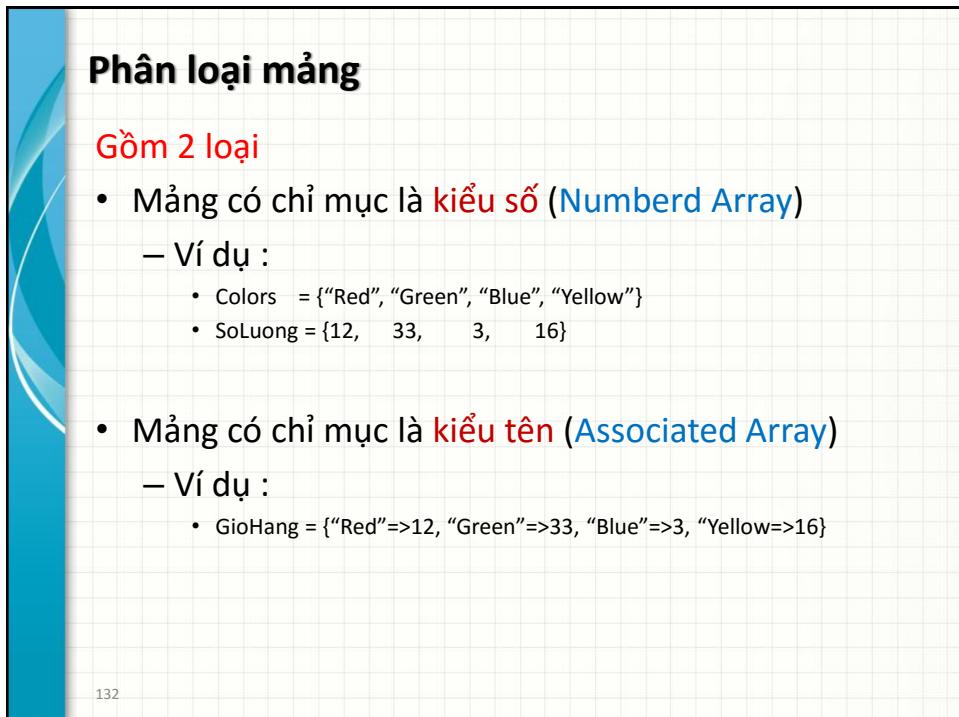
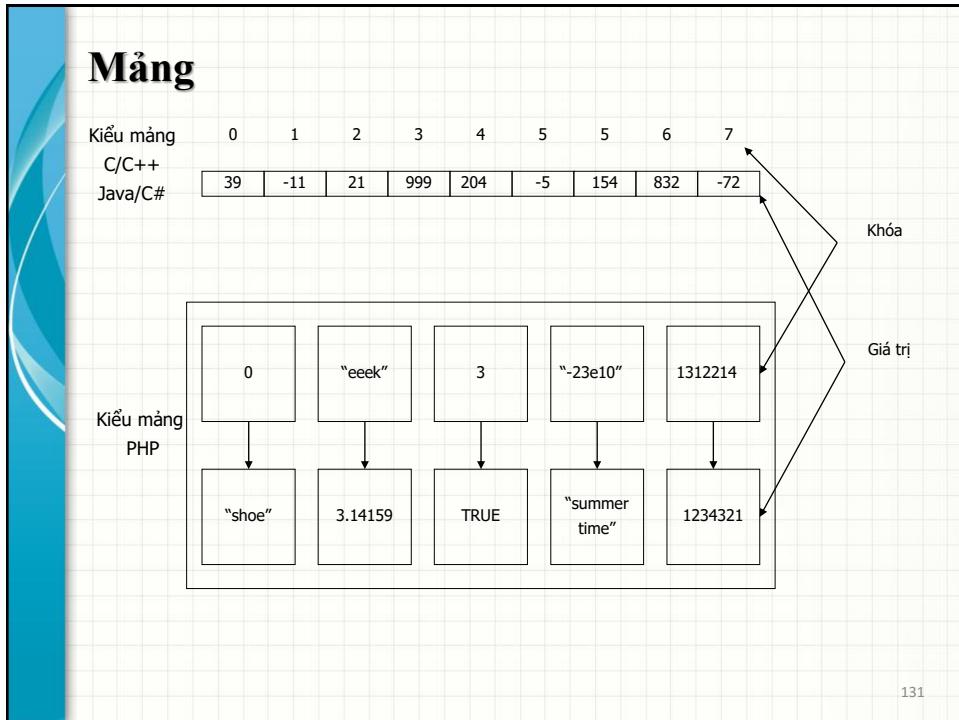
- Mảng
- Chuỗi
- Hàm
- Lập trình hướng đối tượng

129

### 1. Mảng (Array)

- Là một tập các phần tử có **cùng** tính chất
- Mỗi phần tử có thể có
  - **Cùng** kiểu dữ liệu
  - **Khác** kiểu dữ liệu
- Một mảng gồm 2 phần thông tin
  - Thông tin **chỉ mục (khóa)**
  - Thông tin **giá trị** tương ứng với **chỉ mục**
- Mảng đánh chỉ mục từ **0**

130



## Mảng

- Khai báo mảng
  - Không cần khai báo số lượng phần tử
  - Không cần phải khai báo mảng trước khi sử dụng

- Khai báo **tường minh** thông qua **array()**

Ví dụ:

- \$colors = **array**("red", "green", "blue", "yellow");
- echo \$colors[2];

- Khai báo **không tường minh**

Ví dụ:

- \$colors[] = "red";
- \$colors[] = "green";
- \$colors[] = "blue";

133

## Mảng

- Khai báo mảng với mảng **chỉ mục số** có **giá trị** trong 1 khoảng cho trước

**array range(start, end, step)**

- Ví dụ:

- \$number = **range**(0, 10);  
`// $number = array(0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10);`
- \$oddnumber = **range**(1, 10, 2);  
`// $oddnumber = array(1, 3, 5, 7, 9);`
- \$alph = **range**("A", "D");  
`// $alph = array("A", "B", "C", "D");`

134

## Mảng

- Tạo mảng và gán giá trị

`$tên_mảng = array([khóa => ]giá trị, [khóa => ]giá trị,...)`

Trường hợp không định nghĩa các khóa thì mảng sẽ được gán khóa mặc định theo kiểu số nguyên tăng dần **bắt đầu từ 0**

- Ví dụ

`$a=array("Kenny","Maria","Julia","Kenvin");`

`$a= array (name => "Kenny", job => "Teacher",  
age=>"45", email => "webmaster@vietchuyen.com.vn")`

- Tạo mảng từ một mảng có sẵn

`$tên_mảng_mới = $tên_mảng_cũ`

135

## Mảng

- Thêm một phần tử vào mảng

`$tên_mảng[khóa] = giá trị`

- Phần tử luôn được thêm vào cuối mảng
- Nếu khóa đã tồn tại thì không có phần tử nào được thêm
- Trường hợp không chỉ định khóa thì khóa sẽ được chọn bằng khóa có giá trị số nguyên lớn nhất + 1

- Xóa một phần tử khỏi mảng

`unset($tên_mảng[khóa])`

- Đếm số phần tử của mảng

`count($tên_mảng)`

136

## Ví dụ

```
<?php
$arr = array("foo" => "bar", 12 => true);

echo $arr["foo"]; // bar
echo $arr[12];   // 1
?>
```

```
<?php
$arr = array("somearray" => array(6 => 5, 13 => 9, "a" => 42));

echo $arr["somearray"][6];    // 5
echo $arr["somearray"][13];   // 9
echo $arr["somearray"]["a"];  // 42
?>
```

## Mảng

- **Rút trích nội dung**

**\$mảng[key]**

Khi dùng **khóa (key) chuỗi** bên trong một **chuỗi**, không được dùng cặp ‘’ hoặc “ ”, nếu không thì phải đặt truy xuất bên trong cặp {}

- **Ví dụ:**

// sai lỗi cú pháp

```
echo "My PC has a $computer['processor'] processor<br/>\n";
echo "My PC has a $computer["processor"] processor<br/>\n";
```

// đúng cú pháp nhưng không nên dùng

```
echo "My PC has a $computer[processor] processor<br/>\n";
```

// cách dùng tốt nhất

```
echo "My PC has a { $computer['processor'] } processor<br/>\n";
```

## Ví dụ

```
<?php
$arr = array(5 => 1, 12 => 2);

$arr[] = 56; // This is the same as $arr[13] = 56;
              // at this point of the script

$arr["x"] = 42; // This adds a new element to
                  // the array with key "x"

unset($arr[5]); // This removes the element from the array

unset($arr); // This deletes the whole array
?>
```

139

## Mảng

- Rút trích nội dung
  - Sử dụng hàm **list(items)**

Ví dụ:

```
$prod = array("redz", "1", 77, "blue");
list($name, $id, $quantity, $color) = $prod;
// $name="redz", $id="1", $quantity="77", $color="blue"
```

- Sử dụng hàm **extract(array)**

Ví dụ:

```
$prod = array("name"=>"redz", "id"=>"1",
             "quantity"=>77, "color"=>"blue");
extract($prod);
echo "n=$name; id=$id; quan=$quantity; color=$color";

// n=redz; id=1; quan=77; color=red
```

140

## Mảng

- Duyệt mảng với vòng lặp foreach

```
foreach ( $array as $value )  
{    câu lệnh;  
}
```

```
foreach ( $array as $key => $value )
```

```
{    câu lệnh;  
}
```

141

## Ví dụ

```
<?php  
// Create a simple array.  
$array = array(1, 2, 3, 4, 5);  
print_r($array);  
  
// Now delete every item, but leave the array itself intact:  
foreach ($array as $i => $value) {  
    unset($array[$i]);  
}  
print_r($array);  
  
// Append an item (note that the new key is 5, instead of 0).  
$array[] = 6;  
print_r($array);  
  
// Re-index:  
$array = array_values($array);  
$array[] = 7;  
print_r($array);  
?>
```

## Các hàm xử lý mảng

- Các hàm cơ bản

- min(\$mảng)
- max(\$mảng)
- arr\_sum(\$mảng)
- sizeof(\$mảng)
- count(\$mảng)

143

## Các hàm xử lý mảng

- Sắp xếp

**Theo giá trị**

- sort(\$mảng) / asort(\$mảng) // tăng dần
- rsort(\$mảng) / arsort(\$mảng) // giảm dần
- natsort(\$mảng) / natcasesort(\$mảng) // tăng dần, dùng cho chuỗi
- usort(\$mảng, "hàm\_so\_sánh") // tự định nghĩa thứ tự
- uasort(\$mảng, "hàm\_so\_sánh") // tự định nghĩa thứ tự

**Theo khóa**

- ksort(\$mảng) // tăng dần
- krsort(\$mảng) // giảm dần
- uksort(\$mảng, "hàm\_so\_sánh") // tự định nghĩa thứ tự

144

## Các hàm xử lý mảng

- **Sắp xếp**

Tên hàm	Kiểu sắp xếp
sort(\$mang)	Sắp xếp tăng dần theo giá trị (value), khóa (key) thay đổi
asort(\$mang)	Sắp xếp tăng dần theo giá trị (value), khóa (key) vẫn giữ nguyên
rsort(\$mang)	Sắp xếp tăng dần theo giá trị , khóa thay đổi
arsort(\$mang)	Sắp xếp giảm dần theo giá trị, khóa vẫn giữ nguyên
ksort(\$mang)	Sắp xếp tăng dần theo khóa (key)
krsort(\$mang)	Sắp xếp giảm dần theo khóa (key)
usort(\$mang,hàm)	Sắp xếp theo hàm

145

## Các hàm xử lý mảng

- **Sắp xếp**

Ví dụ:

```
$cities=array("Khánh Hòa","Bình Thuận","Lâm Đồng");
```

```
sort($cities);
```

```
print_r($cities);
```

```
// Array ( [0] => Bình Thuận [1] => Khánh Hòa [2] => Lâm Đồng )
```

```
asort($cities);
```

```
print_r($cities);
```

```
// Array ( [1] => Bình Thuận [0] => Khánh Hòa [2] => Lâm Đồng )
```

146

## Các hàm xử lý mảng

- Nối ghép hai mảng

array\_merge(\$mảng1, \$mảng2)

array\_combine(\$mảng1, \$mảng2)

array\_intersect(\$mảng1, \$mảng2)

- Tìm kiếm

array\_search(\$giá\_trị, \$mảng)

147

## Mảng 2 chiều

Ví dụ:

```
<?php
$arrs=array(array(1,2,3,4,5,6,7),
           array(11,12,13,14,15,16,17));
for($i=0;$i<=7;$i++)
{
    for($j=0;$j<=7;$j++)
    {
        $arrs[$i][$j]=10+$i*$j;
        echo $arrs[$i][$j]. "\t";
    }
    echo "<br>";
}
?>
```

148

## Mảng 2 chiều

Ví dụ:

```
$list=array("CNTT"=>array("KTPM"=>"Hang","HTTH"=>"Thuy",
                      "MMT"=>"Nam"),
            "NN"=>array("PD"=>"Thu","DL"=>"Khanh"));
```

```
foreach($list as $khoa=>$value)
{
    //print a heading:
    echo "<h2>$khoa</h2><ul>";
    //print each teacher:
    foreach($value as $gv=>$name)
    {
        echo"<li>$gv - $name</li>\n";
    }
    echo '</ul>';
}
```

149

## 2. Chuỗi (String)

- Một chuỗi là một dãy các ký tự
- Một ký tự giống như một byte
- PHP không giới hạn kích thước kiểu chuỗi, nó chỉ phụ thuộc vào bộ nhớ mà PHP đang chạy.
- Các cách để biểu diễn một chuỗi:
  - Dùng dấu ngoặc đơn ('')
  - Dùng dấu ngoặc kép ("")

150

## Dùng dấu ngoặc đơn ("")

- Đây là cách đơn giản nhất để thể hiện chuỗi, đặt chuỗi cần hiện trong dấu ngoặc đơn
- Để hiển thị dấu nháy đơn trong chuỗi in ra thì dùng ký tự \ (backsplash) trước ký tự '
- Chú ý:
  - Các biến trong dấu ngoặc đơn và các ký tự đặc biệt cho xuống dòng sẽ không được PHP biên dịch **trong dấu ngoặc đơn**.

151

## Ví dụ

```
<?php
$expand = "variable";
echo 'this is a simple string';

echo 'You can also have embedded newlines in
strings this way as it is
okay to do';

// Outputs: Arnold once said: "I'll be back"
echo 'Arnold once said: "I\'ll be back"';

// Outputs: You deleted C:\*.*?
echo 'You deleted C:\\*.*?';

// Outputs: You deleted C:\*.*?
echo 'You deleted C:\\*.*?';

// Outputs: This will not expand: \n a newline
echo 'This will not expand: \\n a newline';

// Outputs: Variables do not $expand $either
echo 'Variables do not $expand $either';
?>
```

152

## Dùng dấu ngoặc kép ("")

- PHP sẽ biên dịch các biến và các ký tự đặc biệt ở trong dấu ngoặc kép.

Sequence	Meaning
\n	linefeed (LF or 0x0A (10) in ASCII)
\r	carriage return (CR or 0x0D (13) in ASCII)
\t	horizontal tab (HT or 0x09 (9) in ASCII)
\v	vertical tab (VT or 0x0B (11) in ASCII) (since PHP 5.2.5)
\f	form feed (FF or 0x0C (12) in ASCII) (since PHP 5.2.5)
\\\	backslash
\\$	dollar sign
\"	double-quote
\{0-7\}{1,3}	the sequence of characters matching the regular expression is a character in octal notation
\{0-9A-Fa-f\}{1,2}	the sequence of characters matching the regular expression is a character in hexadecimal notation

153

## Một số hàm xử lý chuỗi

### Các xử lý cơ bản

- **trim(\$chuỗi, ['ký tự muốn cắt'])**
- **ltrim(\$chuỗi, ['ký tự muốn cắt'])**
- **rtrim(\$chuỗi, ['ký tự muốn cắt'])**
- **strlen(\$chuỗi)**
- **substr(\$chuỗi, \$vị trí, \$chiều\_dài)**
- **strtoupper(\$chuỗi)**
- **strtolower(\$chuỗi)**
- **ucfirst(\$chuỗi); //đổi ký tự đầu tiên thành chữ hoa**
- **ucwords(\$chuỗi); //đổi ký tự đầu mỗi từ thành chữ hoa**
- **iconv(mã nguồn, mã đích, \$chuỗi)**

154

## Ví dụ hàm trim()

```
<?php

$text = "\t\tThese are a few words :) ... ";
$binary = "\x09Example string\x0A";
$hello = "Hello World";
var_dump($text, $binary, $hello);

print "\n";

$trimmed = trim($text);
var_dump($trimmed);

$trimmed = trim($text, " \t.");
var_dump($trimmed);

$trimmed = trim($hello, "Hdle");
var_dump($trimmed);

// trim the ASCII control characters at the beginning and end of $binary
// (from 0 to 31 inclusive)
$clean = trim($binary, "\x00..\x1F");
var_dump($clean);

?>
```

155

## Vd: hàm iconv() -> convert chuỗi theo loại mã nào đó

```
<?php
$text = "This is the Euro symbol '€'.";

echo 'Original : ', $text, PHP_EOL;
echo 'TRANSLIT : ', iconv("UTF-8", "ISO-8859-1//TRANSLIT", $text), PHP_EOL;
echo 'IGNORE   : ', iconv("UTF-8", "ISO-8859-1//IGNORE", $text), PHP_EOL;
echo 'Plain    : ', iconv("UTF-8", "ISO-8859-1", $text), PHP_EOL;

?>
```

The above example will output something similar to:

```
Original : This is the Euro symbol '€'.
TRANSLIT : This is the Euro symbol 'EUR'.
IGNORE   : This is the Euro symbol ''.
Plain    :
Notice: iconv(): Detected an illegal character in input string in .\iconv-example.php on line 7
This is the Euro symbol '
```

## Một số hàm xử lý chuỗi

- Tìm kiếm
  - `strpos($chuỗi, $chuỗi_con, [$vị_trí_bắt đầu])`
  - `strrpos($chuỗi, $chuỗi_con, [$vị_trí_bắt đầu])`
- So sánh
  - `strcmp($chuỗi_1, $chuỗi_2)`
  - `strncmp($chuỗi_1, $chuỗi_2, $chiều_dài)`
  - `strcasecmp($chuỗi_1, $chuỗi_2)`
  - `strncasecmp($chuỗi_1, $chuỗi_2, $chiều_dài)`
  - `strnatcmp($chuỗi_1, $chuỗi_2)`
  - `strnatcasecmp($chuỗi_1, $chuỗi_2)`
- Tách hay kết hợp chuỗi
  - `explode("Tập ký tự", $chuỗi);`
  - `implode("Tập ký tự", $mảng);`

157

## Ví dụ hàm explode và implode

- Tách chuỗi, đưa phần tử vào mảng  
`explode("Tập ký tự", $chuỗi);`
- Đưa phần tử của mảng vào chuỗi  
`implode("Tập ký tự", $mảng);`

Ví dụ:

```
<?php
$str="Sun-Mon-Tue-Wed-Thu-Fri-Sat";
$arr=explode("-", $str);
var_dump($arr);
$str=implode(", ",$arr);
var_dump($str);
?>
```

158

## Lưu ý khi so sánh chuỗi

- Không nên dùng toán tử Equal (==) mà nên dùng toán tử Identical (====). Bởi vì == không so sánh kiểu dữ liệu và có thể dẫn đến sai lệch trong việc so sánh
- Ví dụ:

```
<?php
    $input = 0; echo ('defaultpassword' == $input) ? 'true' : 'false';
//return true
?>
```

- Dùng toán tử so sánh === để so sánh 2 chuỗi

```
<?php echo ('1e3' == '1000') ? 'true' : 'false'; ?> //return true
```

```
<?php echo ('1e3' === '1000') ? 'true' : 'false'; ?> //return false
```

## Xử lý kiểu datetime

- Ví dụ:

```
<html>
<head>
<title>Date and Time</title>
</head>
<body>
<h4>Ngay hien tai</h4>
<?php
    echo date("j-S-F-Y");
    echo "<br>";
    echo date("M/Y");
    echo "<br>";
    echo "Days of ".date("M")." is ".date("t");
    echo "<br>";
?>
</body>
</html>
```

## Bảng các tham số của hàm date

Code	Điển giải
a	Buổi sáng/Chiều bằng hai ký tự thường am/pm.
A	Buổi sáng/Chiều bằng hai ký tự hoa AM/PM.
B	Định dạng thời gian Swatch Internet, bạn có thể tham khảo <a href="http://swatch.com/internetttime/internetttime.php3">http://swatch.com/internetttime/internetttime.php3</a> .
d	Day (01-31) trong tháng với hai số, nếu ngày 1-9 sẽ có kèm số 0.
D	Day (Mon-Sun) trong tuần với 3 ký tự.
F	Tháng (January-December) trong năm với tên tháng đầy đủ dạng text.
g	Hour (1-12) trong ngày 1 hoặc 2 số (không kèm 0 nếu giờ từ 1-9).
G	Hour (0-23) trong ngày 1 hoặc 2 số (không kèm 0 nếu giờ từ 0-9).
h	Hour (01-12) trong ngày 2 số (kèm 0 nếu giờ từ 01-09).
H	Hour (00-23) trong ngày 2 số (kèm 00 nếu giờ từ 00-09).
i	Minutes (01-59) đã trôi qua (kèm 0 nếu phút từ 00-59).
j	Day (1-31) 1 hoặc 2 số (không kèm 0 nếu ngày từ 1-9).
I	Day (Monday-Sunday) trong tuần dạng text.

## Ví dụ sử dụng hàm strpos()

```
<?php
$mystring = 'abc';
$findme   = 'a';
$pos = strpos($mystring, $findme);

// Note our use of ===. Simply == would not work as expected
// because the position of 'a' was the 0th (first) character.
if ($pos === false) {
    echo "The string '$findme' was not found in the string '$mystring'";
} else {
    echo "The string '$findme' was found in the string '$mystring'";
    echo " and exists at position $pos";
}
?>
```

### 3. Hàm

- Mục đích xây dựng hàm
  - Tái sử dụng lại những đoạn mã giống nhau.
  - Tăng tính mềm dẻo, nhất quán trong ứng dụng, thời gian xây dựng và thiết kế ứng dụng.
- Các lợi ích
  - Giảm được thời gian và chi phí
  - Giúp tăng độ tin cậy của ứng dụng vì các function và module đã được xây dựng, kiểm tra và sử dụng tốt cho các ứng dụng trước cũng có thể được sử dụng lại cho các ứng dụng khác với mục đích tương tự
  - Tạo sự nhất quán trong thiết kế và xây dựng ứng dụng

163

### Định nghĩa Hàm

- Định nghĩa

<?php

```
function tên_hàm([tham số 1, tham số 2, ...])
{
    [thân hàm ...]
    [return;]
}
```

?>

- Gọi hàm

Nhập tên\_hàm (...) và cung cấp đầy đủ các tham số cần thiết trong cặp dấu ()

164

## Ví dụ Hàm

```
<?php
function testing($a) {
    echo "Tham số là $a";
} //end testing
//Gọi hàm
testing(123);
testing("abc");
?>
```

165

## Hàm

- **Kết thúc và trả kết quả**

Lệnh **return** dùng để kết thúc và trả kết quả cũng như quyền điều khiển lại cho nơi đã gọi hàm. Nếu không có lệnh **return** thì mặc định hàm trả về giá trị NULL.

Muốn trả về **hơn một giá trị** thì phải **dùng mảng**

```
<?php
function binh_phuong($a)
{
    $ketqua = $a * $a;
    return $ketqua;
} //end function

echo binh_phuong(2);
?>
```

166

## Hàm

- **Truyền tham số**

Mặc định các tham số được truyền vào bên trong hàm theo phương pháp **tham trị**. Trường hợp **muốn thay đổi trực tiếp trên các tham số truyền** thì người ta dùng phương pháp **tham chiếu**, thêm dấu & trước tên tham số (khi định nghĩa) cũng như tên biến được truyền làm tham số (khi gọi hàm)

```
function functionName(parameters, &parameters)
{
    function-body
}
```

167

## Hàm

- **Truyền tham số**

Ví dụ:

```
<?php
function changeToBoldText($text){
    $text=<b>".$text."</b>";
}
function changeToRedText(&$text){
    $text=<font color='red'>".$text."</font>";
}
function changeToBoldRedText(&$text){
    changeToRedText($text);
    changeToBoldText($text);
}
$mytext="Truong dai hoc Nha Trang";
changeToBoldRedText($mytext);
echo $mytext;
?>
```

168

## Hàm

- **Tham số có giá trị mặc định**

Tương tự cách khai báo và gán giá trị đầu tiên cho biến, thông thường **loại tham số này nên đặt cuối trong danh sách** tham số

```
function functionName(parameters, parameter=value)
{
    function-body
}
```

Khi gọi hàm **nếu bỏ trống** tại vị trí tham số có giá trị mặc định thì mặc nhiên **giá trị mặc định được dùng cho tham số đó**

169

## Hàm

- **Tham số có giá trị mặc định**  
– Ví dụ

<pre>1 &lt;?php 2     function tinhGiaSauVAT(\$GiaSP, \$VungMien){ 3         switch (\$VungMien){ 4             case "Quan 1": 5                 echo \$GiaSP+\$GiaSP*0.1; 6                 break; 7             case "Quan 2": 8                 echo \$GiaSP+\$GiaSP*0.25; 9                 break; 10            case "Quan 5": 11                echo \$GiaSP+\$GiaSP*0.33; 12                break; 13            default: 14                echo \$GiaSP+\$GiaSP*0.11; 15            } 16        } 17 18        tinhGiaSauVAT(150, "Quan 1"); 19    ?&gt;</pre>	<pre>1 &lt;?php 2     function tinhGiaSauVAT(\$GiaSP, \$VungMien='Quan 5'){ 3         switch (\$VungMien){ 4             case "Quan 1": 5                 echo \$GiaSP+\$GiaSP*0.1; 6                 break; 7             case "Quan 2": 8                 echo \$GiaSP+\$GiaSP*0.25; 9                 break; 10            case "Quan 5": 11                echo \$GiaSP+\$GiaSP*0.33; 12                break; 13            default: 14                echo \$GiaSP+\$GiaSP*0.11; 15            } 16        } 17 18        tinhGiaSauVAT(150); 19    ?&gt;</pre>
------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------

170

## Hàm

- **Hàm có số lượng tham số không xác định**
  - Khai báo danh sách tham số rỗng ()
  - Sử dụng các hàm sau để lấy danh sách các tham số:

*func\_num\_args():* số lượng tham số khi hàm được gọi

*func\_get\_arg(i):* giá trị các tham số thứ i được truyền (bắt đầu từ 0)

*func\_get\_args():* danh sách tất cả các tham số

171

## Hàm

- **Hàm có số lượng tham số không xác định**

Ví dụ:

```
<?php
function makecoffee($type = "Trung Nguyen")
{
    return "Making a cup of $type.\n";
}
```

```
echo makecoffee();
echo makecoffee(null);
echo makecoffee("VinaCafe");
```

?>

Kết quả:

*Making a cup of Trung Nguyen  
Making a cup of  
Making a cup of VinaCafe*

172

## Hàm

- Hàm có số lượng tham số không xác định

Ví dụ:

```
<?php
    function foo()
    {
        $numargs = func_num_args();
        echo "Number of arguments: $numargs\n";
    }

    foo(1, 2, 3);
?>
```

173

## Hàm

- **Biến tĩnh**

Thêm từ khóa **static** khi khai báo biến

Được khởi tạo (và gán giá trị) một lần đầu tiên duy nhất trong suốt quá trình thực thi của script

- **Sử dụng biến toàn cục**

Khai báo lại biến toàn cục với từ khóa **global** (bên trong hàm) để có thể sử dụng được biến toàn cục này bên trong hàm

174

## Hàm

Ví dụ:

```
function Test ()  
{  
    static $a = 0;  
    echo $a;  
    $a++;  
}
```

```
$a=1;  
$b=2;  
function sum(){  
    global $a, $b;  
    $b=$a+$b;  
}  
sum();  
echo $b;
```

- Với khai báo như trên, \$a sẽ không mất đi giá trị sau khi gọi hàm Test() mà \$a sẽ được tăng lên 1 sau mỗi lần gọi hàm Test().

175

## Hàm

- **Hàm biến**

Khi một **biến kiểu chuỗi** được khai báo và gán giá trị trùng khớp với tên một hàm được định nghĩa thì tên biến đó có thể được dùng như một cách gọi hàm khác với cách gọi hàm bình thường bằng tên hàm.

Một số hàm không thể dùng như hàm biến

- echo
- print
- var\_dump
- print\_r
- isset
- unset
- is\_null
- is\_type

176

## Ví dụ

```
<?php
function foo() {
    echo "In foo()<br />\n";
}
function bar($arg = "")
{
    echo "In bar(); argument was '$arg'.<br />\n";
}
function echoit($string)
{
    echo $string;
}
$func = 'foo';
$func(); // This calls foo()
$func = 'bar';
$func('test'); // This calls bar()
$func = 'echoit';
$func('test'); // This calls echoit()
?>
```

177

## Sử dụng lại các Hàm

- PHP cung cấp các phương thức rất đơn giản và hữu dụng cho phép sử dụng lại các đoạn chương trình đã viết là **require()** và **include()**
- Hai phương thức này dùng để chèn file vào trang PHP.
- Các file được chèn có thể chứa các đoạn chương trình PHP, thẻ HTML, văn bản, hàm và class.

178

**03\_IncReq\_MainTestInc.php**

```

1 <?php
2     echo "<div style='width:100%; height:200px;
3             background-color:yellow;'>";
4     echo "    <table style='width:100%;'>;
5         <tr>";
6         echo "            <td style='width:30%'> LOGO </td>";
7         echo "            <td style='width:70%'> SLOGAN </td>";
8     echo "        </tr>";
9     echo "    </table>";
10    echo "</div>";
11 ?>
12 
```

**03\_IncReq\_MainTest.php**

```

1 <!DOCTYPE HTML PUBLIC "-//IWS//DTD HTML 4.01 Transitional//EN">
2 <html>
3     <head>
4         <meta http-equiv="Content-Type"
5             content="text/html; charset=UTF-8">
6         <title></title>
7     </head>
8     <body>
9         <?php include("03_IncReq_MainTestInc.php"); ?>
10
11     <div style="width:100%; height:400px;
12         background-color: lightblue;">
13         <center>
14             <h3>PHP & MySQL</h3>
15         </center>
16     </div>
17     </body>
18 </html>
19 
```

## Require

- Cách sử dụng
- Các loại tập tin được dùng trong require()
- Thẻ PHP và require()
- Dùng require() cho các template

## Require

- **Cách sử dụng**

- **Đặc điểm**

- Tất cả nội dung bên trong file được chèn vào trang sẽ được biên dịch.
    - Muốn sử dụng đoạn chương trình đã được viết sẵn ở vị trí nào trong trang thì dùng require() để chèn file chứa đoạn chương trình này ở vị trí đó.

- **Cú pháp**

```
require("tên_và_đường_dẫn_của_file");
```

181

## Require

- **Các loại tập tin được dùng trong require()**

- Các câu lệnh PHP được viết trong các file .html sẽ không thực hiện được.

- Các câu lệnh PHP trong các file được dùng để chèn phải có kiểu file thích hợp như .inc hay .php để các câu lệnh này có thể biên dịch được khi thực thi.

- **Chú ý:**

- PHP không quy định cách đặt tên file đối với file được chèn vào bằng require() → có thể đặt tên file tùy ý.

- Dùng require() để chèn file → nội dung của file sẽ trở thành một phần trong trang PHP.

182

## Require

- **Thẻ PHP và require()**
  - Các lệnh PHP cần phải được bao bởi cặp thẻ PHP **<?php** và **?>**
  - Nếu không có thẻ PHP **<?php** và **?>** khi viết thì các lệnh PHP này sẽ trở thành dạng văn bản hoặc HTML và không thể thực thi được.

183

## Require

- **Thẻ PHP và require()**

– Ví dụ:

```
// functions.inc
<?php
function bold($string)
{
    echo "<b>" . $string . "</b>\n";
}
?>
```

```
// index.php
<html>
<head>
    <title>Simple Function Call</title>
</head>
<body bgcolor="#ffffff">
<?
require "functions.inc";
bold("this is bold");

$myString = "this is bold";
bold($myString);
?>
</body></html>
```

## Require

- Dùng require() cho các template
  - Nếu ứng dụng Web có cùng một mẫu thiết kế và chỉ khác nhau về nội dung bên trong → tạo ra một template và khai báo các biến trình bày dữ liệu bên trong template.
  - Khi template này được thực thi, tất cả các biến này sẽ có giá trị và trình bày như một định dạng template được sử dụng nhiều lần.  
→ Dùng require() để chèn template vào các trang cần thiết.

185

## Require

- Dùng require() cho các template
  - Ví dụ:
    - Trong ứng dụng web phần header và footer thường được hiển thị ở hầu hết các trang → tạo ra trang **header.inc** để chứa định dạng và nội dung phần đầu trang, và trang **footer.inc** để chứa định dạng và nội dung phần cuối trang.
    - Trang nào của ứng dụng có sử dụng header và footer thì dùng require() để chèn hai trang này vào.

186

## Require

- Dùng require() cho các template

– Ví dụ:

- Điểm đặc biệt và quan trọng nhất của việc kết hợp này là sau khi đã thực hiện nhiều trang có chèn các trang **header.inc** và **footer.inc**, ta có thể dễ dàng thay đổi trang **header.inc** và trang **footer.inc** này. Chỉ cần lưu lại các thay đổi thì tất cả các trang đã chèn các trang này sẽ tự động thay đổi.

187

## Include

- Cách sử dụng
- require\_once() và include\_once()

188

## Include

- **Cách sử dụng**

- **Đặc điểm:**

- include() cũng có cách sử dụng tương tự như require().
- Điểm khác nhau của include() và require(): khi nội dung bị lỗi thì dùng require() sẽ xuất hiện thông báo lỗi trong khi đó dùng include() sẽ chỉ xuất hiện cảnh báo.
- Trong những file có dùng require() thì không nên sử dụng các cấu trúc điều khiển vì sẽ không hiệu quả hoặc sử dụng include() thay cho require().

- **Cú pháp:**

include("tên\_và\_đường\_dẫn\_của\_file");

189

## Include

- **Cách sử dụng**

- **Ví dụ:**

```

<?php
if($a==1)
{
    include("tinh_tong.php");
}
else
{
    include("tinh_hieu.php")
}
?>
```

190

## Include

- **require\_once() và include\_once()**
  - require\_once() và include\_once() là hai dạng biến đổi của require() và include()
  - Mục đích sử dụng: đảm bảo cho việc chèn file chỉ được thực hiện duy nhất một lần.
  - Chức năng:
    - Các hàm này trở nên hữu ích khi bắt đầu sử dụng chúng để chèn các thư viện và các hàm vào
    - Sử dụng các cấu trúc này giúp tránh được việc chèn cùng một hàm hay thư viện lần thứ hai, bởi vì khi khai báo lại một hàm đã được xây dựng sẽ phát sinh lỗi.

191