

A light gray world map is visible in the background, centered on the Atlantic Ocean. The continents of North America, South America, Europe, Africa, and Asia are outlined in a darker shade of gray.

Chương 5.1

WEBSERVER

ThS. Thiều Thanh Quang phú

Mục tiêu

- ❖ Hiểu và sử dụng được ngôn ngữ lập trình PHP
- ❖ Vận dụng để xây dựng trang web bằng PHP

Chương 5.1. WebServer

1.1. Giới thiệu WebServer

1.2. Cài đặt WebServer

1.3. Các thao tác trên PhpMyAdmin

1.1. Giới Thiệu WebServer

- ❖ Là 1 máy tính có cài đặt phần mềm dịch vụ Web.
- ❖ WebServer phải có cấu hình “mạnh” (dung lượng lớn, tốc độ cao,...) và phải chạy liên tục.
- ❖ Tất cả các WebServer đều có một địa chỉ IP (IP address) và một tên miền (Domain name).
 - ✚ Ví dụ: Trang Google có tên miền: <http://www.google.com.vn> , địa chỉ IP: <http://74.125.128.94>

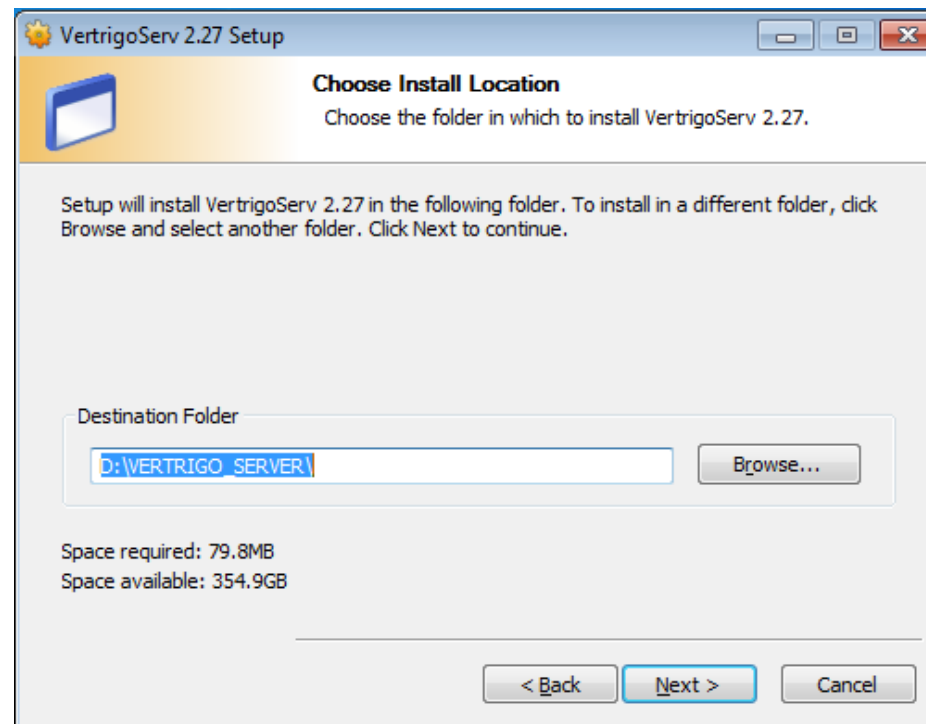
1.1. Giới thiệu WebServer

- ❖ Có 2 loại phần mềm Server phổ biến:
 - ✚ IIS Server (Internet Information Services): chỉ chạy trên Windows, thường dùng cho HTML, ASP, ASP.NET.
 - ✚ Apache Server: mã nguồn mở, chạy trên các hệ điều hành, thường dùng cho HTML, PHP,...
- ❖ Có nhiều phần mềm tạo Apache Server: Vertrigo, WampServer, XAMPP, AppServ, USBWebServer, ... Khi cài đặt sẽ có những dịch vụ chính sau:
 - ✚ Apache Server
 - ✚ Bộ biên dịch PHP
 - ✚ CSDL MySQL
 - ✚ Hệ quản trị CSDL phpMyAdmin.

1.2. Cài đặt WebServer

❖ Các bước chuẩn bị với Vertrigo:

- # **B1:** Tải Vertrigo về từ địa chỉ <http://vertrigo.sourceforge.net> (phiên bản mới nhất)
- # **B2:** Cài Vertrigo vào máy tính (nên cài vào thư mục gốc của ổ đĩa).



1.2. Cài đặt WebServer

❖ Các bước chuẩn bị với Vertrigo:

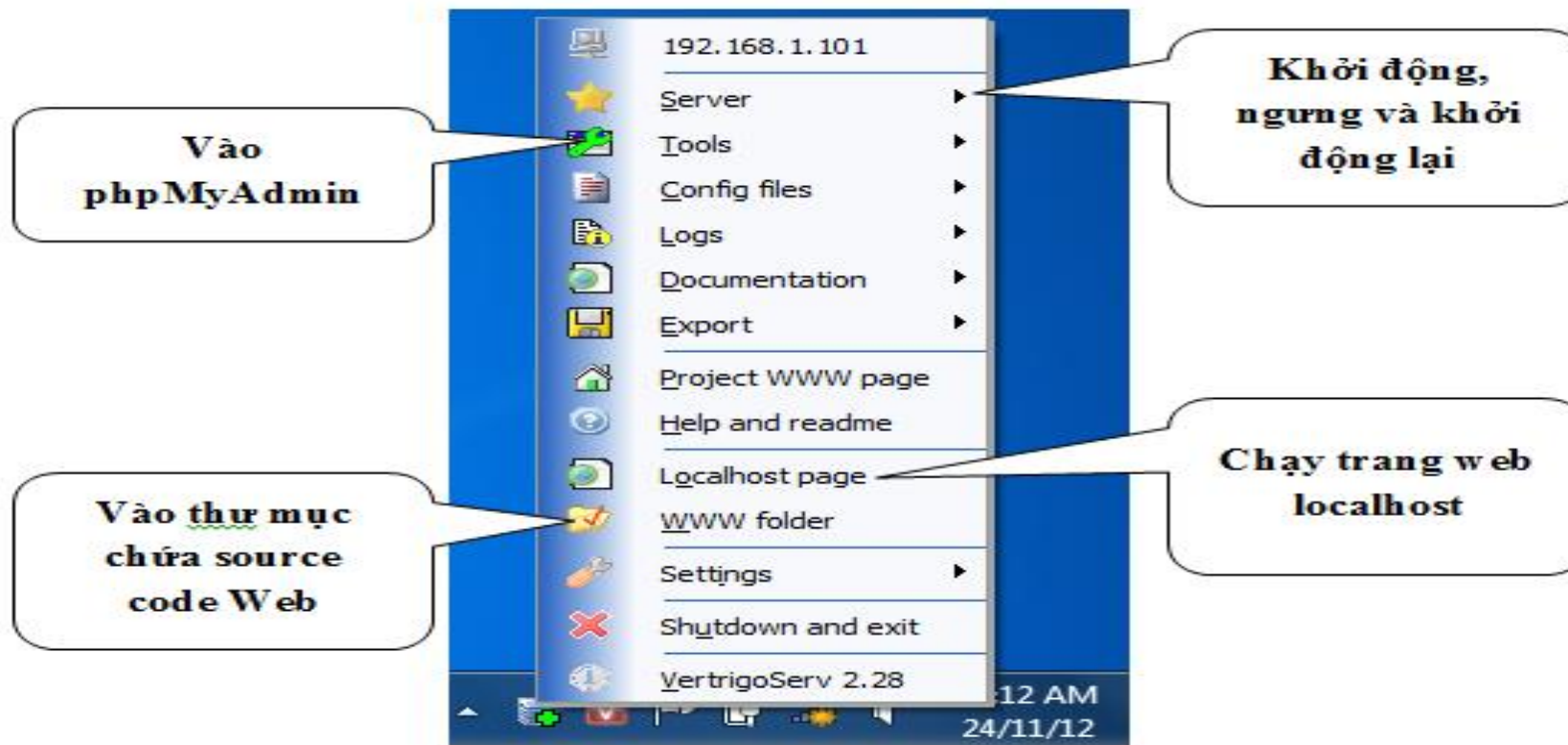
✚ **B3:** Khởi động phần mềm bằng biểu tượng ngoài màn hình nền, đầu tiên sẽ xuất hiện cửa sổ như hình bên dưới:



1.2. Cài đặt WebServer

❖ Các bước chuẩn bị với Vertrigo:

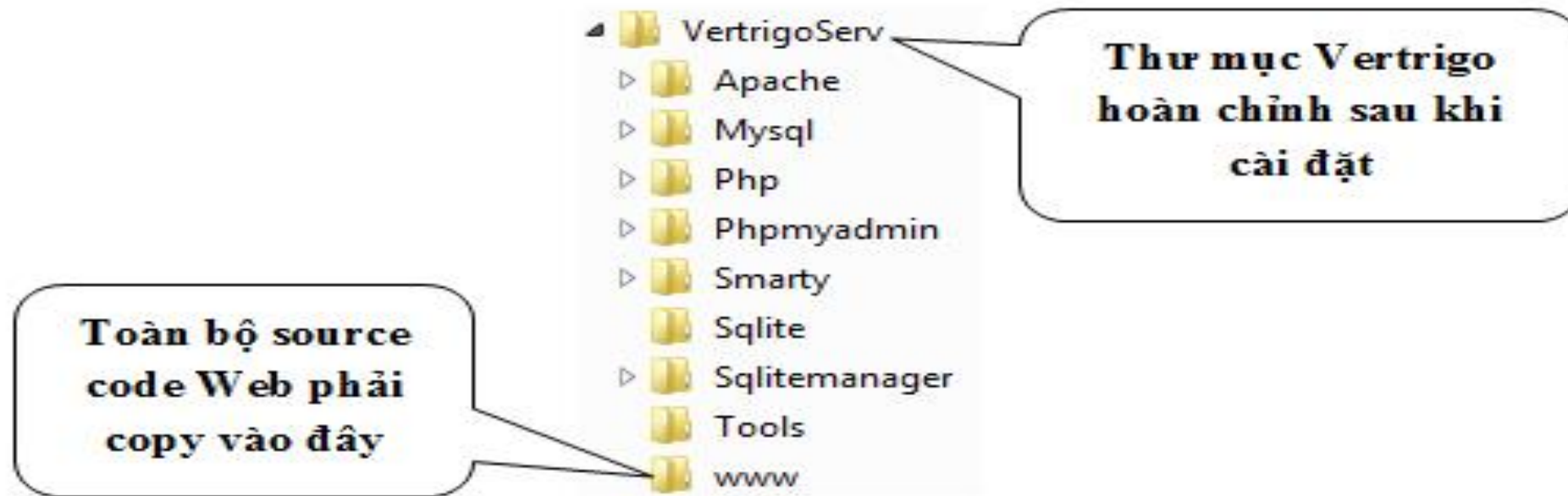
✚ **B4.** Sau khi chạy Server, dưới khay đồng hồ sẽ có một biểu tượng của Vertrigo, click vào đó sẽ hiện ra một menu với nhiều mục chọn.



1.2. Cài đặt WebServer

❖ Các bước chuẩn bị với Vertrigo:

- ✚ **B5:** Chạy trang Server bằng cách mở trình duyệt và gõ vào tên miền: <http://localhost> hoặc địa chỉ IP: <http://127.0.0.1>
- ✚ **B6:** Xóa hết các tập tin trong thư mục www sau khi cài đặt xong Vertrigo.



1.3. Các thao tác trên phpMyAdmin

- ❖ Là phần mềm mã nguồn mở trên nền Web, được tích hợp trong Vertrigo, WampServer, AppServ, XAMPP,...
- ❖ Giao diện thân thiện, dễ dàng quản lý và thực thi các câu lệnh MySQL.
- ❖ Được phát triển từ năm 1998 (GNU General Public License).
- ❖ Độc lập với nền tảng hệ điều hành (Linux/UNIX, MacOS, Windows,...).

1.3. Các thao tác trên phpMyAdmin

- ❖ Sau khi cài Vertrigo (WampServer/AppServ/XAMPP), khởi động chương trình bằng cách Click chuột phải vào biểu tượng trên thanh taskbar, chọn:
 - ✚ Tools\PhpMyAdmin.
 - ✚ Hoặc mở trình duyệt web và gõ vào thanh địa chỉ:
<http://localhost/phpmyadmin>
hoặc <http://127.0.0.1/phpmyadmin>
- ❖ Trong trường hợp dùng Vertrigo thì đăng nhập với thông tin mặc định:
root/vertrigo

1.3. Các thao tác trên phpMyAdmin

❖ Giao diện phpMyAdmin

127.0.0.1 / localhost | phpMyAdmin 3.3.9.2 - Mozilla Firefox

File Edit View History Bookmarks Tools Help

http://127.0.0.1/phpmyadmin/

127.0.0.1 / localhost | phpMyAdmin 3...

phpMyAdmin

- information_schema (37)
- mysql (24)
- performance_schema (17)
- phpmyadmin (9)
- test

Please select a database

localhost

Databases SQL Status VariablesCharsets Engines Privileges Replication Processes Export Import Synchronize

Actions

- Change password
- Log out

MySQL localhost

- Create new database
- MySQL connection collation: utf8_unicode_ci

Interface

- Language: English
- Theme / Style: Original
- Custom color: Reset
- Font size: 82%

MySQL

- Server: localhost via TCP/IP
- Server version: 5.5.10
- Protocol version: 10
- User: root@localhost
- MySQL charset: UTF-8 Unicode (utf8)

Web server

- Apache/2.2.17 (Win32) PHP/5.3.6
- MySQL client version: mysqlnd 5.0.8-dev - 20102224 - \$Revision: 308673 \$
- PHP extension: mysqli

phpMyAdmin

- Version information: 3.3.9.2
- Documentation
- Wiki
- Official Homepage
- [ChangeLog] [Git] [Lists]

Done

Menu chức năng.

Đổi mật khẩu.

Tạo CSDL mới.

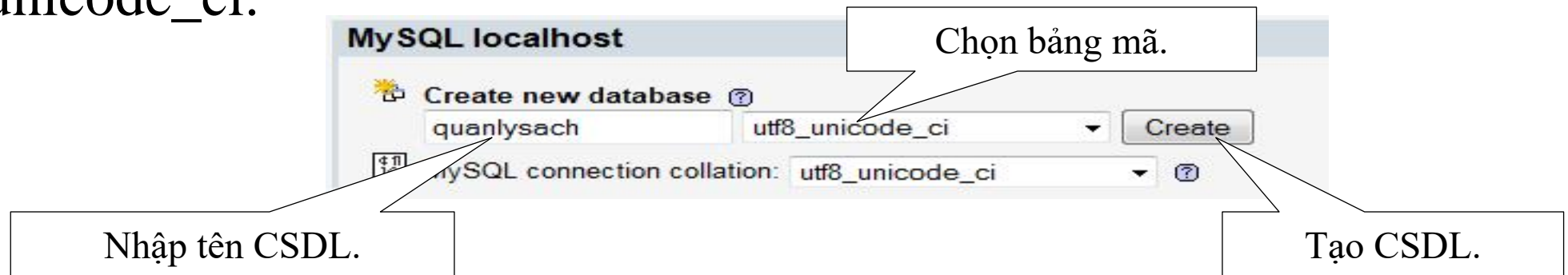
Danh sách các Bảng.

Thông tin về MySQL, Web server và phpMyAdmin.

1.3. Các thao tác trên phpMyAdmin

❖ Tạo CSDL

- ✚ Để CSDL có thể lưu trữ được tiếng Việt, nên chọn bảng mã là utf8_unicode_ci.



- ✚ Lưu ý: Tên CSDL phải

- Đảm bảo quy tắc về ký tự (không nên có dấu, không chứa khoảng trắng, không chứa ký tự đặc biệt,...).
- Không trùng tên với CSDL đã có.

1.3. Các thao tác trên phpMyAdmin

❖ Tạo bảng

- ✚ Chọn CSDL cần tạo bảng ở danh sách bên trái, sau đó nhập vào tên bảng cần tạo.



- ✚ Lưu ý: Tên Bảng phải

- Đảm bảo quy tắc về tên giống tên CSDL.
- Không trùng với tên bảng đã có trong cùng 1 CSDL.
- Nên đặt tên bảng có “tiếp đầu ngữ” để dễ quản lý

1.3. Các thao tác trên phpMyAdmin

The screenshot displays the phpMyAdmin interface for a database named 'quanlysach'. The left sidebar shows the database name and a message 'No tables found in database.' The main area shows the structure of the table 'tbl_loaisach' with two fields: 'MaLoai' and 'TenLoai'. The 'MaLoai' field is configured as an integer (INT) with a length of 10, set as the primary key (PRIMARY) and auto-incrementing. The 'TenLoai' field is configured as a variable-length string (VARCHAR) with a length of 200. Both fields use the 'utf8_unicode_ci' collation. The 'Table comments' section at the bottom contains the text 'Bảng loại sách'. The 'Storage Engine' is set to 'InnoDB' and the 'Collation' is 'utf8_unicode_ci'. The 'Save' button is visible at the bottom right.

Annotations in Vietnamese:

- Tên trường. (Field Name)
- Kiểu dữ liệu. (Data Type)
- Độ dài dữ liệu. (Data Length)
- Bảng mã. (Collation)
- Chú thích trường. (Field Comment)
- Bảng mã. (Collation)
- Dạng lưu trữ. (Storage Engine)
- Tạo Bảng. (Create Table)
- Chú thích Bảng. (Table Comment)
- Khóa. (Key)

1.3. Các thao tác trên phpMyAdmin

❖ Xem dữ liệu

The screenshot shows the phpMyAdmin interface. On the left, the 'Database' dropdown is set to 'quanlysach (4)', and the table 'tbl_nhaxuatban' is selected. The main area displays the table structure and data. The SQL query is: `SELECT * FROM `tbl_nhaxuatban` LIMIT 0, 30`. The table data is shown in a grid with columns 'MaNhaXB' and 'TenNhaXB'.

Server: localhost Database: quanlysach Table: tbl_nhaxuatban

Showing rows 0 - 6 (7 total, Query took 0.0004 sec)

SQL query: `SELECT * FROM `tbl_nhaxuatban` LIMIT 0, 30`

Showing 30 row(s) starting from record # 0 in horizontal mode and repeat headers after 100 cells

Sort by key: None

| | | | MaNhaXB | TenNhaXB |
|--------------------------|--|--|---------|-------------------------|
| <input type="checkbox"/> | | | GD | Giáo Dục |
| <input type="checkbox"/> | | | HCM | Tổng Hợp TP.Hồ Chí Minh |
| <input type="checkbox"/> | | | NHV | Hội Nhà Văn |
| <input type="checkbox"/> | | | PN | Phụ Nữ |
| <input type="checkbox"/> | | | TN | Thanh Niên |
| <input type="checkbox"/> | | | VH | Văn Học |
| <input type="checkbox"/> | | | VHTT | Văn Hóa Thông Tin |

Check All / Uncheck All With selected:

cần

1.3. Các thao tác trên phpMyAdmin

❖ Tìm kiếm dữ liệu

The screenshot displays the phpMyAdmin web interface. On the left sidebar, the 'Database' dropdown is set to 'quanlysach (4)', and the table 'tbl_nhaxuatban' is selected. The main panel shows the 'SQL' tab with the query: `SELECT * FROM `tbl_nhaxuatban` WHERE 1`. The 'Fields' list on the right includes 'MaNhaXB' and 'TenNhaXB'. At the bottom, there are options to bookmark the query and checkboxes for 'Let every user access this bookmark' and 'Replace existing bookmark of same name'. A 'Go' button is located at the bottom right.

Server: localhost ▶ Database: quanlysach ▶ Table: tbl_nhaxuatban

Browse Structure SQL Search Insert Export Import Operations Empty Drop

Run SQL query/queries on database **quanlysach**: ?

```
SELECT * FROM `tbl_nhaxuatban` WHERE 1
```

Fields

- MaNhaXB
- TenNhaXB

Bookmark this SQL query:

☐ Let every user access this bookmark
☐ Replace existing bookmark of same name

[Delimiter :] ☒ Show this query here again



Chương 5.2

CƠ BẢN VỀ PHP

ThS. Thiều Thanh Quang phú

Chương 5.2. Cơ bản về PHP

- 2.1. Giới thiệu PHP
- 2.2. Cấu trúc của PHP
- 2.3. Hiện thị dữ liệu trong PHP
- 2.4. Chạy chương trình đầu tiên
- 2.5. Biến và Hằng trong PHP
- 2.6. Các kiểu dữ liệu trong PHP
- 2.7. Các phép toán
- 2.8. Biểu thức chính quy

2.1. Giới thiệu PHP

- ❖ PHP viết tắt là Personal Home Page hay Hypertext Preprocessor
- ❖ PHP (Hypertext Preprocessor) là ngôn ngữ script ở phía server.
- ❖ PHP hỗ trợ nhiều cơ sở dữ liệu (MySQL, Informix, Oracle, Sybase, Solid, PostgreSQL,...)
- ❖ Tập tin PHP:
 - ✚ Có thể chứa text, các thẻ HTML và các đoạn script nhưng kết quả khi trả về trình duyệt là HTML
 - ✚ Có phần mở rộng là **".php"**

2.1. Giới thiệu PHP

- ❖ PHP: Rasmus Lerdorf in 1994
- ❖ PHP 2 (1995): script xử lý trên server. Hỗ trợ CSDL, Upload File, khai báo biến, mảng, hàm đệ quy,...
- ❖ PHP 3 (1998): Hỗ trợ ODBC, đa hệ điều hành, giao thức email (SNMP, IMAP), bộ phân tích mã PHP (parser)
- ❖ PHP 4 (2000): Trở thành một thành phần độc lập cho các webserver. Parse đổi tên thành Zend Engine
- ❖ PHP 5 (2005): Bổ sung Zend Engine II hỗ trợ lập trình OOP, XML, SOAP cho Web Services, SQLite.
- ❖ PHP 5.4.7 (2012)
- ❖ PHP 6: một dự án thử nghiệm nhưng không trở thành một phiên bản hoàn chỉnh.
- ❖ PHP7 (2015)

2.1. Giới thiệu PHP



❖ Các hệ thống xây dựng bằng PHP

- ✚ Hệ thống chuyên về Quản trị nội dung, cổng thông tin (CMS – Content Management System / Portals)
- ✚ Hệ thống chuyên về Diễn đàn (Forum)
- ✚ Hệ thống chuyên về Blog: WordPress, Textpattern, Nucleus CMS, LifeType, Serendipity, Dotclear, Zomplog, FlatPress, ...
- ✚ Hệ thống về thương mại điện tử (eCommerce): Magento, Zen Cart, OpenCart, osCommerce, PrestaShop, AlegroCart, Freeway, ...
- ✚ Hệ thống về đào tạo trực tuyến (LCMS–Learning Course Management System) Moodle, ATutor, eFront, Dokeos, Docebo, Interact, DrupalEd, ILIAS, Open Conference Systems, ...

2.1. Giới thiệu PHP

- ❖ Để bắt đầu với PHP và MySQL, ta cần chuẩn bị:
 - ✚ Cài đặt web server (IIS, Apache, ...)
 - ✚ Cài PHP
 - ✚ Cài MySQL
- ❖ Download:
 - ✚ Apache: httpd.apache.org/download.cgi
 - ✚ PHP: www.php.net/downloads.php
 - ✚ MySQL: www.mysql.com/downloads/index.html
- ❖ Đơn giản hơn, có thể download các phần mềm WAMP, Vertrigo, XAMPP...

2.2. Cấu trúc của PHP

- ❖ Một khối lệnh PHP có thể đặt bất cứ nơi nào trong trang web
- ❖ Mỗi câu lệnh kết thúc bởi dấu ;
- ❖ Cú pháp gần giống C++, Perl
- ❖ Phân biệt chữ hoa, thường,...
- ❖ Ghi chú trong PHP:
 -  // ghi chú đơn
 -  /* đoạn ghi chú */

2.2. Cấu trúc của PHP

❖ Những code PHP trong trang HTML:

| Thẻ mở | Thẻ đóng | Ghi chú |
|-------------------------|----------|---|
| <? | ?> | Cần cấu hình server cho phép hỗ trợ shorthand-support → ít dùng |
| <?php | ?> | Thường dùng |
| <script language="php"> | <script> | ít dùng |

❖ Ví dụ:

```
<html>
<body>
<?php
//In ra chuỗi "Hello World"
echo "Hello World";
?>
</body>
</html>
```

2.3. Hiện thị dữ liệu trong PHP

❖ Hiện thị văn bản

- ✚ Sử dụng phương thức echo: `echo "noidung";`
- ✚ Hoặc `echo ("noidung");`
- ✚ Hoặc `echo 'noi dung';`

```
<?php
    echo "Nguyễn Văn A";
    echo "<br>";
    echo 'Lập Trình Web';
?>
```

❖ Hiện thị giá trị của biến

```
<?php
    $name = "Nguyễn Văn A";
    $year = 2002;
    echo $name;
    echo "<hr>";
    echo $year;
?>
```

2.3. Hiện thị dữ liệu trong PHP

❖ Hiện thị phần tử HTML

- ✚ Dùng lệnh echo để hiển thị các phần tử HTML.
- ✚ Đặt các phần tử HTML vào bên trong cặp dấu nháy kép hoặc cặp dấu nháy đơn.

```
<?php
    echo "<h1>HTML</h1>";
    echo '<h2>Ngôn ngữ lập trình PHP</h2>';
    echo "<h3>Hướng dẫn học
    <h4>Lập trình web</h4>
    <p><b><i><u>từ cơ bản đến nâng cao</u></i></b></p>";
?>
```

HTML

Ngôn ngữ lập trình PHP

Hướng dẫn học

Lập trình web

từ cơ bản đến nâng cao

```
<?php
    echo "<p>Tài <b style='color:blue'>liệu</b> học HTML</p>";
    echo '<p>Tài <b style="color:green">liệu</b> học CSS</p>';
?>
```

Tài **liệu** học HTML

Tài **liệu** học CSS

2.3. Hiện thị dữ liệu trong PHP

- ❖ Nối các phần tử lại với nhau:
 - ✚ Đặt dấu chấm nằm giữa hai nội dung.

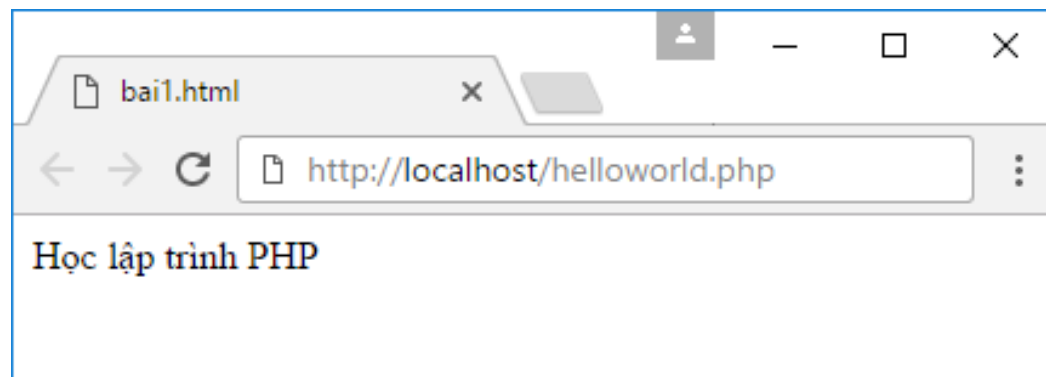
```
<?php
    $text = " sống tại Cần thơ";
    echo "Tôi tên là " . "Nguyễn Thành Nhân" . " sinh năm " . 1993 . $text;
?>
```

2.4. Chạy chương trình đầu tiên

- ❖ Đầu tiên tạo một file **helloworld.php** trong thư mục **www** của Vertrigo và viết code như sau:

```
<?php
    echo "Học lập trình PHP";
?>
```

- ❖ Khởi động Vertrigo lên và chạy với đường dẫn **http://localhost/helloworld.php**
- ❖ Kết quả:



2.5. Biến và Hằng trong PHP

❖ Biến (*variable*):

- ✚ Biến phải được bắt đầu bằng ký tự \$.
- ✚ Tên biến có thể bao gồm các ký tự (A..Z, a..z), ký số (0..9), _,
- ✚ Phân biệt chữ hoa, chữ thường
- ✚ Không được bắt đầu bằng ký số (0..9)
- ✚ Không khai báo kiểu dữ liệu, kiểu dữ liệu tự động được khởi tạo khi gán giá trị cho biến.

```
<?php
    $php= 'hello'; //đúng
    $_php='hello'; //đúng
    $5php='hello'; //sai vì bắt đầu bằng số;
    $-php='hello'; //sai vì bắt đầu bằng -
?>
```

2.5. Biến và Hằng trong PHP

❖ Biến (*variable*):

- ✚ Gán giá trị cho biến

`$bien = value`

- ✚ Hiện thị giá trị biến ra trình duyệt

```
<?php
    $a= 'Học lập trình PHP';
    echo "Bạn vừa tham gia " + $a;
?>
```

2.5. Biến và Hằng trong PHP

❖ Phạm vi của biến:

- ✚ Global (toàn cầu, toàn cục)
- ✚ Local (cục bộ)
- ✚ Static (tĩnh)

❖ Global (toàn cầu, toàn cục)

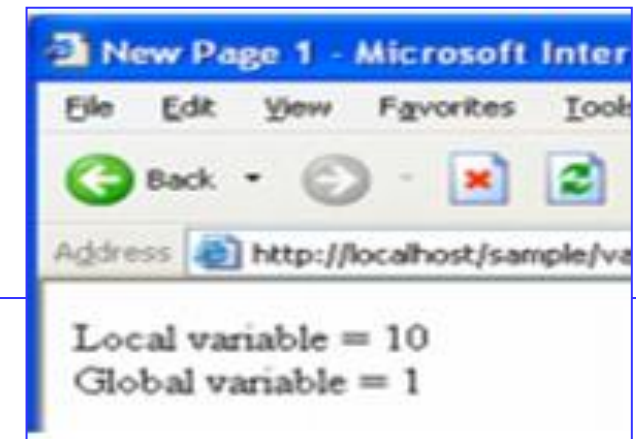
- ✚ Biến được khai báo bên ngoài hàm
- ✚ Sử dụng từ khóa global trước khai báo biến

```
<?php
$a = 1;

function printLocalVariable() {
    $a = 10;
    echo "Local variable = " . $a . "<br>";
}

function printGlobalVariable() {
    global $a;
    echo "Global variable = " . $a . "<br>";
}

printLocalVariable();
printGlobalVariable();
?>
```



2.5. Biến và Hằng trong PHP

❖ Static (tĩnh)

- ✚ Khi một hàm hoàn thành việc thực thi, biến static sẽ không bị xóa.
- ✚ Sau mỗi lần hàm được gọi thì biến vẫn có thông tin về giá trị mà nó chứa từ lần cuối cùng hàm được gọi.

012

```
<?php  
  
    function myTest() {  
        static $a = 0;  
        echo $a;  
        $a++;  
    }  
  
    myTest();  
    myTest();  
    myTest();  
  
?>
```

2.5. Biến và Hằng trong PHP

❖ Hằng (*variable*):

✚ Hằng là một loại biến nhưng không thể thay đổi giá trị được.

✚ Cú pháp khai báo hằng:

`define('Tên_hằng', Giá_trị);`

➤ define: là cú pháp mặc định để tạo phương thức trong PHP.

➤ Tên_Hằng: là tên của hằng.

➤ Giá_Trị: là giá trị của hằng.

```
1 <?php
2 /* Tạo một hằng số có tên là SDT và gán giá trị cho nó là 0909090909 */
3 define('SDT', '0909090909');
4 echo SDT; // xuất ra màn hình giá trị của hằng.
5 ?>
```

2.6. Các kiểu dữ liệu trong PHP

❖ Các kiểu dữ liệu cơ bản như sau:

- ✚ String (chuỗi)
- ✚ Integer (số nguyên)
- ✚ Float (số thực, số dấu phẩy động)
- ✚ Boolean (đúng/sai)
- ✚ Array (mảng)
- ✚ Object (đối tượng)
- ✚ NULL (giá trị NULL)
- ✚ Resource (tài nguyên)

2.6. Các kiểu dữ liệu trong PHP

❖ Kiểu int

✚ Ví dụ:

```
<?php
    $a = 1234; // hệ số thập phân
    $a = -123; // số âm
    $a = 0123; // Bát phân
    $a = 0x1A; // Hệ thập lục
    $a = 0b11111111; // Hệ nhị phân
?>
```

2.6. Các kiểu dữ liệu trong PHP

❖ Kiểu int

✚ Ép dữ liệu sang kiểu int:

```
1  <?php
2  $a = 'a123'; // biến $a có giá trị là chuỗi 'a123'
3  $a = (int)$a; // chuyển $a sang kiểu INT
4  echo $a; // kết quả xuất ra màn hình là số 0
5  ?>
```

✚ Kiểm tra dữ liệu có phải kiểu int sử dụng 1 trong 2 phương thức sau. Kết quả trả về giá trị **True** nếu là kiểu int, **False** nếu không phải kiểu int.

- is_int(\$bien)
- hoặc is_integer(\$bien)

2.6. Các kiểu dữ liệu trong PHP

❖ Kiểu boolean

✚ Chỉ chứa 2 giá trị là TRUE hoặc FALSE

✚ Ví dụ:

```
1  <?php
2  $is_admin = false;
3  ?>
```

✚ Ép kiểu sang boolean (bool và boolean là 2 từ khóa cùng một ý nghĩa)

```
1  <?php
2  $bool = 1; // biến $bool là kiểu int
3  $bool = (bool)$bool; // lúc này biến $bool sẽ có kiểu boolean
4  // Hoặc
5  $bool = (boolean)$bool; // lúc này biến $bool sẽ có kiểu boolean
6  ?>
```

✚ Kiểm tra 1 biến có phải kiểu boolean không dùng hàm is_bool(\$bien)

2.6. Các kiểu dữ liệu trong PHP

❖ Kiểu số thực (float, double)

```
1 <?php
2 $a = 1.234; // Kiểu số thực
3 ?>
```

✚ Ép dữ liệu sang kiểu số thực:

```
1 <?php
2 $a = 123; // biến $a kiểu int
3 $a = (float)$a; // Biến $a lúc này kiểu số thực (float)
4 $a = (double)$a; // Biến $a lúc này kiểu số thực (double)
5 ?>
```

✚ Để kiểm tra một biến phải kiểu số thực dùng hàm `is_float($bien)` và `is_double($bien)`

2.6. Các kiểu dữ liệu trong PHP

❖ Kiểu chuỗi

- ✚ Kiểu chuỗi tồn tại ở hai dạng là String và Char.
- ✚ Để khai báo chuỗi thì giá trị của chuỗi phải được đặt trong cặp dấu ngoặc ' hoặc ''.

```
<?php
    $string = 'Lập trình PHP'; //đúng
    $string = "Học CSS";       // đúng
    $string = Học Javascript;   // sai
?>
```


2.6. Các kiểu dữ liệu trong PHP

- ❖ Khác nhau giữa dấu nháy kép “ ” và dấu nháy đơn ‘ ’
 - + Dấu nháy kép “ ” : Sẽ hiển thị luôn giá trị của biến.
 - + Dấu nháy đơn ‘ ’ : Không hiển thị giá trị của biến.

```
<?php
```

```
    $name = "Nguyễn Văn A";  
    $text_1 = "Tên của tôi là $name";  
    $text_2 = 'Tên của tôi là $name';  
    echo $text_1;  
    echo "<hr>";  
    echo $text_2;
```

```
?>
```

Tên của tôi là Nguyễn Văn A

Tên của tôi là \$name

2.6. Các kiểu dữ liệu trong PHP

❖ Kiểu chuỗi

- # Nối chuỗi: dùng dấu chấm “.”
- # Phương thức xử lý chuỗi thông dụng

| | | | |
|----------------------|--------------------------|-------------------------|------------------|
| <code>printf</code> | <code>trim</code> | <code>strtolower</code> | |
| <code>str_pad</code> | <code>str_replace</code> | <code>strtoupper</code> | |
| <code>strlen</code> | <code>substr</code> | <code>strcasecmp</code> | <code>...</code> |

2.6. Các kiểu dữ liệu trong PHP

❖ Các hàm xử lý chuỗi

```
<?php

$txt1="Hello World";
$txt2="1234";

//Nối chuỗi
echo $txt1 . " " . $txt2 . "<br>";

//Tính độ dài chuỗi
echo strlen("Hello world!") . "<br>";

//Tìm vị trí chuỗi con
echo strpos("Hello world!","world")

?>
```

```
Hello World 1234
12
6
```

2.6. Các kiểu dữ liệu trong PHP

❖ Các hàm xử lý chuỗi

```
<?php
$text = "HTML and CSS";
$a = strlen($text); //biến $a sẽ có giá trị là 12
$b = strlen("JavaScript"); //biến $b sẽ có giá trị là 10
$c = strlen('MySQL & PHP'); //biến $c sẽ có giá trị là 11
?>
```

```
<?php
$text_1 = "123456789";
$text_2 = strrev($text_1);
//hàm strrev đảo ngược thứ tự
//các ký tự trong chuỗi
echo $text_1;
echo "<hr>";
echo $text_2;
?>
```

123456789

987654321

2.6. Các kiểu dữ liệu trong PHP

❖ Kiểu mảng

✚ Là một danh sách các phần tử có cùng, hoặc không cùng kiểu dữ liệu.

✚ Khai báo mảng

➤ Cách 1:

```
<?php $array = array(); ?>
```

```
<?php $array = array('giá trị 1','giá trị 2'); ?>
```

➤ Cách 2:

```
<?php $array = []; ?>
```

```
<?php $array = ['giá trị 1','giá trị 2']; ?>
```

2.6. Các kiểu dữ liệu trong PHP

❖ Kiểu mảng

```
<?php
    $mobile = array("HTC", "Nokia", "SungSung", "LG", "Blackberry");
?>
```

| | Phần tử 1 | Phần tử 2 | Phần tử 3 | Phần tử 4 | Phần tử 5 |
|---------|-----------|-----------|-----------|-----------|------------|
| Chỉ số | 0 | 1 | 2 | 3 | 4 |
| Giá trị | HTC | Nokia | SamSung | LG | Blackberry |

✚ Hiển thị mảng.

- Phương thức **print_r()** in ra các phần tử và vị trí của nó trong mảng.

```
<?php
    $bien = array('CSS', 'HTML', 'PHP');
    print_r($bien);
?>
```

2.6. Các kiểu dữ liệu trong PHP

❖ Kiểu mảng

✚ Hiển thị mảng.

- In ra từng phần tử của mảng

```
<?php
    $bien = array('CSS', 'HTML', 'PHP');
    echo $bien[0]; //CSS
    echo $bien[1]; //HTML
    echo $bien[2]; // PHP
?>
```

2.6. Các kiểu dữ liệu trong PHP

❖ Kiểu mảng

✚ Thêm phần tử vào mảng

➤ Cú pháp:

`$TenMang['key'] = 'value';`

- Tenmang: Là tên của mảng.
- Key: là trị số mảng, nếu không điền thì php sẽ tự thêm vào cuối mảng.
- Value: là giá trị phần tử.

```
<?php
    $bien = array('CSS', 'PHP');
    $bien[2] = HTML;
    // kết quả: ['CSS', 'PHP', 'HTML']
?>
```


2.6. Các kiểu dữ liệu trong PHP

❖ Kiểu mảng

✚ Sửa phần tử trong mảng

➤ Cú Pháp:

<?php \$mang['index'] = 'giá trị mới'; ?>

- mang: Là tên của mảng.
- index: là trị số mảng, nếu không điền thì php sẽ tự lấy cuối mảng.
- giá trị mới: là giá trị phần tử.

2.6. Các kiểu dữ liệu trong PHP

❖ Kiểu mảng

✚ Có các loại mảng:

- Numeric array: phân biệt các giá trị bằng chỉ số
- Associative array: phân biệt các giá trị bằng tên
- Multidimensional array: mảng đa chiều

2.6. Các kiểu dữ liệu trong PHP

❖ Kiểu mảng

✚ Numeric array: phân biệt các phần tử mảng bằng chỉ số mảng

➤ Tạo numeric array

- Chỉ số được gán tự động

```
$names = array("Peter", "Quagmire", "Joe");
```

- Chỉ số gán bằng tay

```
$names[0] = "Peter";  
$names[1] = "Quagmire";  
$names[2] = "Joe";
```

2.6. Các kiểu dữ liệu trong PHP

❖ Kiểu mảng

✚ Associative array: phân biệt các phần tử bằng tên đi kèm với giá trị

✚ Ví dụ:

```
$prices = array("pen"=>1000, "pencil"=>500);
```

Hoặc

```
$prices['pen'] = 1000;  
$prices['pencil'] = 500;
```

Cách dùng

```
<?php  
$prices['pen'] = 1000;  
$prices['pencil'] = 500;  
echo "Pen costs " . $prices['pen'] . " VND.";  
?>
```

2.6. Các kiểu dữ liệu trong PHP

❖ Kiểu mảng

✚ Multidimensional array: mảng đa chiều

```
<?php
$profile = array(
    'name' => 'Nguyễn Văn A',
    'year' => 2003,
    'hobbies' => array(
        'xem phim',
        'nghe nhạc',
        'chăm sóc cây cảnh',
        array(
            'game_1' => 'GTA 5',
            'game_2' => 'Võ Lâm Truyền Kỳ'
        )
    ),
    'city' => 'Cần Thơ'
);
echo $profile['name'] . "<br>";
echo $profile['hobbies'][1] . "<br>";
echo $profile['hobbies'][3]['game_2'];
?>
```

Nguyễn Văn A
nghe nhạc
Võ Lâm Truyền Kỳ

2.6. Các kiểu dữ liệu trong PHP

❖ Kiểu mảng

foreach(): truy xuất các phần tử của mảng

```
foreach ($arr as $key => $value)
{
    echo $key.", ";
    echo $value."<br />";
}
```

\$key => \$value tương ứng là \$chỉ số=>\$giá trị.

2.6. Các kiểu dữ liệu trong PHP

❖ Kiểu mảng

each(): truy xuất các phần tử của mảng

```
while ($item = each ($arr) )  
{  
    echo $item['key'] . ", ";  
    echo $item['value'] . "<br />";  
}
```

Phương thức each() trả về phần tử kế tiếp của mảng. Từ khóa key và value phải là số nguyên.

2.6. Các kiểu dữ liệu trong PHP

❖ Kiểu mảng

- ✚ list(): truy xuất các phần tử của mảng
- ✚ Hàm list(\$k,\$v) tách cặp giá trị (key,value) của phần tử có “chỉ số” là \$key ra hai biến \$k và \$v tương ứng.

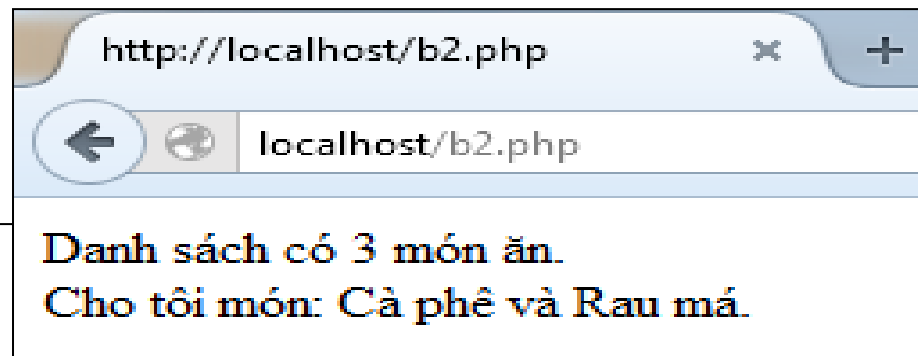
```
<?php
    $arr = array(10);
    for($i=0; $i<10; $i++)
        $arr[$i] = rand(0,100);
    while(list($k,$v) = each($arr))
    {
        echo $k.", ";
        echo $v."<br />";
    }
?>
```

```
0, 71
1, 95
2, 6
3, 96
4, 72
5, 38
6, 7
7, 36
8, 72
9, 90
```


2.6. Các kiểu dữ liệu trong PHP

❖ Kiểu mảng

✚ Ví dụ:



```
<html>
  <body>
    <?php
      $dinner = array( 'Rau má', 'Sinh tố bơ', 'Cà phê' );
      $dishes = count( $dinner );
      print "Danh sách có $dishes món ăn.";
      sort( $dinner );
      echo "</br> Cho tôi món: $dinner[0] và $dinner[1].";
    ?>
  </body>
</html>
```

2.6. Các kiểu dữ liệu trong PHP

❖ Kiểu mảng

✚ Các phương thức thường dùng:

| Hàm | Mô tả |
|-----------------------|--|
| <code>sort()</code> | - Sắp xếp các phần tử mảng theo thứ tự tăng dần của giá trị (chỉ dành cho mảng được lập chỉ mục) |
| <code>rsort()</code> | - Sắp xếp các phần tử mảng theo thứ tự giảm dần của giá trị (chỉ dành cho mảng được lập chỉ mục) |
| <code>asort()</code> | - Sắp xếp các phần tử mảng theo thứ tự tăng dần của giá trị (chỉ dành cho mảng kết hợp) |
| <code>arsort()</code> | - Sắp xếp các phần tử mảng theo thứ tự giảm dần của giá trị (chỉ dành cho mảng kết hợp) |
| <code>ksort()</code> | - Sắp xếp các phần tử mảng theo thứ tự tăng dần của khóa. |
| <code>krsort()</code> | - Sắp xếp các phần tử mảng theo thứ tự giảm dần của khóa. |

```
<?php
    $data = array("B", "D", "C", "A");
    sort($data);
    echo "<pre>";
    print_r($data);
    echo "</pre>";
?>
```

```
Array
(
    [0] => A
    [1] => B
    [2] => C
    [3] => D
)
```

2.6. Các kiểu dữ liệu trong PHP

❖ Kiểu Đối tượng (Object)

- ✚ Một lớp là một khuôn mẫu cho các đối tượng, và một đối tượng là một thể hiện của một lớp.
- ✚ Khi các đối tượng riêng lẻ được tạo, chúng kế thừa tất cả các thuộc tính và hành vi từ lớp, nhưng mỗi đối tượng sẽ có các giá trị khác nhau cho các thuộc tính.

```
<?php
class Mobile{
    public $color;
    public $model;
    public function __construct($color,$model){
        $this->color = $color;
        $this->model = $model;
    }
    public function message(){
        return "Tôi dùng {$this->model}, nó có màu {$this->color}";
    }
}

$myMobile = new Mobile("Xám", "Nokia 8.1");
echo $myMobile -> message();
echo "<br>";
$myMobile = new Mobile("Đen", "SamSung Galaxy 3");
echo $myMobile -> message();
?>
```

2.7. Các phép toán

❖ Toán tử

- ✚ Số học (Arithmetic)
- ✚ Gán (Assignment)
- ✚ So sánh (Comparison)
- ✚ Logic (Logical)

2.7. Các phép toán

❖ Toán tử

✚ Số học (Arithmetic)

| Operator | Description | Example | Result |
|----------|------------------------------|---------------------|-------------|
| + | Addition | x=2 x+2 | 4 |
| - | Subtraction | x=2 5-x | 3 |
| * | Multiplication | x=4 x*5 | 20 |
| / | Division | 15/5 5/2 | 3 2.5 |
| % | Modulus (division remainder) | 5%2 10%8 10%2 | 1 2 0 |
| ++ | Increment | x=5 x++ | x=6 |
| -- | Decrement | x=5 x-- | x=4 |

2.7. Các phép toán

❖ Toán tử

✚ Gán (Assignment)

| Operator | Example | Is The Same As |
|----------|------------|----------------|
| = | $x = y$ | $x = y$ |
| += | $x += y$ | $x = x + y$ |
| -= | $x -= y$ | $x = x - y$ |
| *= | $x *= y$ | $x = x * y$ |
| /= | $x /= y$ | $x = x / y$ |
| %= | $x \% = y$ | $x = x \% y$ |

2.7. Các phép toán

❖ Toán tử

✚ So sánh (Comparison)

| Operator | Description | Example |
|--------------------|-----------------------------|------------------------------------|
| <code>==</code> | is equal to | <code>5==8</code> returns false |
| <code>!=</code> | is not equal | <code>5!=8</code> returns true |
| <code>></code> | is greater than | <code>5>8</code> returns false |
| <code><</code> | is less than | <code>5<8</code> returns true |
| <code>>=</code> | is greater than or equal to | <code>5>=8</code> returns false |
| <code><=</code> | is less than or equal to | <code>5<=8</code> returns true |

2.7. Các phép toán

❖ Toán tử

✚ Logic (Logical)

| Operator | Description | Example |
|----------|-------------|---|
| && | and | x=6 y=3 (x < 10 && y > 1) returns true |
| | or | x=6 y=3 (x==5 y==5) returns false |
| ! | not | x=6 y=3 !(x==y) returns true |

2.8. Biểu thức chính quy

❖ Biểu thức chính quy (Regular Expression, viết tắt là RegEx)

- ✚ Là một chuỗi các ký tự tạo thành một “mẫu tìm kiếm”, dùng để mô tả nội dung cần tìm.
- ✚ Biểu thức chính quy có thể là một ký tự, hoặc một mẫu phức tạp hơn.

```
$exp = "/bieuthuchinhquy/i";
```

- ✚ Dấu / là dấu phân cách; Nó có thể là bất kỳ ký tự nào (không phải là chữ cái, chữ số, dấu khoảng trắng), dấu gạch chéo ngược \. Dấu phân cách được sử dụng phổ biến nhất là dấu gạch chéo /, nhưng khi mẫu của có chứa dấu gạch chéo thì bạn nên chọn các dấu phân cách khác như # hoặc ~ để thay thế.
- ✚ **bieuthuchinhquy** là mẫu đang được tìm kiếm
- ✚ **i** là một bộ ngữ chỉ việc tìm kiếm không phân biệt chữ in hoa hay chữ thường.

2.8. Biểu thức chính quy

- ❖ Các “bổ nghĩa” dùng để mô tả thêm cách thức tìm kiếm trong chuỗi.

| | |
|---|---|
| i | - Thực hiện tìm kiếm không phân biệt chữ in hoa hay chữ thường. |
| m | - Thực hiện tìm kiếm nhiều dòng (các mẫu tìm kiếm phần đầu hoặc phần cuối của một chuỗi sẽ khớp với phần đầu hoặc phần cuối của mỗi dòng) |
| u | - Cho phép đối sánh chính xác các mẫu được mã hóa UTF-8. |

- ❖ Các mẫu biểu thức chính quy:

✚ Cặp dấu ngoặc [] được sử dụng để tìm một loạt các ký tự.

| | |
|--------|--|
| [abc] | - Tìm một ký tự từ các ký tự tùy chọn bên trong cặp dấu ngoặc. |
| [^abc] | - Tìm bất kỳ ký tự nào không phải là một trong các ký tự tùy chọn trong cặp dấu ngoặc. |
| [0-9] | - Tìm một ký tự trong phạm vi 0 - 9. |

2.8. Biểu thức chính quy

❖ Các phương thức của biểu thức chính quy.

| | |
|--------------------------------------|---|
| preg_match() | - Tìm “kết quả đầu tiên” trùng khớp với mẫu cần tìm. |
| preg_match_all() | - Tìm “tất cả kết quả” trùng khớp với mẫu cần tìm. |
| preg_replace() | - Thay thế các kết quả trùng khớp với mẫu cần tìm bằng một chuỗi chỉ định. |
| preg_filter() | - Trả về một chuỗi (hoặc một mảng) với các kết quả khớp với mẫu đã được thay thế, nhưng chỉ khi các kết quả khớp được tìm thấy. |
| preg_grep() | - Trả về một mảng chỉ chứa các phần tử từ đầu vào khớp với mẫu đã cho. |
| preg_last_error() | - Trả về mã lỗi cho biết lý do cuộc gọi biểu thức chính quy gần đây nhất không thành công. |
| preg_replace_callback() | - Cho trước một biểu thức & lệnh gọi lại, trả về một chuỗi trong đó tất cả các kết quả khớp của biểu thức được thay thế bằng chuỗi con được trả về bởi hàm gọi lại. |
| preg_replace_callback_array() | - Trả về một chuỗi hoặc một mảng các chuỗi trong đó các kết quả khớp của một tập các biểu thức chính quy được thay thế bằng giá trị trả về của một hàm gọi lại. |
| preg_split() | - Ngắt chuỗi thành một mảng bằng cách sử dụng các kết quả khớp của biểu thức chính quy làm dấu phân cách. |
| preg_quote() | - Thoát khỏi những ký tự có ý nghĩa đặt biệt trong biểu thức chính quy bằng cách đặt một dấu gạch chéo ngược phía trước chúng. |

2.8. Biểu thức chính quy

❖ Các ký tự Meta

| Metacharacter | Mô tả | Ví dụ |
|---------------|---|---|
| . | Khớp với bất kỳ ký tự đơn nào trừ dòng mới | /./ sẽ khớp với bất cứ thứ gì có 1 ký tự duy nhất |
| ^ | Khớp với phần đầu hoặc chuỗi bắt đầu với ký tự nào đó | /^PH/ sẽ khớp với bất kỳ chuỗi nào bắt đầu bằng PH |
| \$ | Khớp vào pattern ở cuối chuỗi | /com\$/ khớp với chuỗi kết thúc là com (google.com) |
| * | Khớp với bất kỳ ký tự nào | /com*/ sẽ khớp với computer, communication, compare ... |
| + | Yêu cầu các ký tự đứng trước xuất hiện ít nhất 1 lần | /goo+gle/ sẽ khớp với google |
| \ | Sử dụng để ký tự sau nó là nguyên thủy | /google+\.com/ sẽ coi dấu chấm là dấu chấm theo nghĩa đen |
| [...] | Lớp ký tự | /[abc]/ sẽ khớp với abc |
| a-z | Khớp với các ký tự thường từ a-z | /a-z/ sẽ khớp với các ký tự thường từ a-z |
| A-Z | Khớp với các ký tự Hoa từ A-Z | /A-Z/ |
| 0-9 | Khớp bất kỳ số nào trong khoảng từ 0 đến 9 | /0-9/ |

2.8. Biểu thức chính quy

❖ Ví dụ

```
<?php
$my_url = "https://agu.edu.vn";
if (preg_match("/agu/", $my_url))
{
    echo "the url $my_url contains agu";
}
else
{
    echo "the url $my_url does not contain agu";
}
?>
```

the url https://agu.edu.vn contains agu

2.8. Biểu thức chính quy

❖ Ví dụ

```
<?php
    $my_email = "name@company.com";
    if (preg_match("/^[a-zA-Z0-9._-]+@[a-zA-Z0-9-]+\.[a-zA-Z.]{2,5}$/", $my_email)) {
        echo "$my_email is a valid email address";
    }
    else
    {
        echo "$my_email is NOT a valid email address";
    }
?>
```

```
name@company.com is a valid email address
```

2.8. Biểu thức chính quy

❖ Giải thích ý nghĩa pattern: `"[/^[a-zA-Z0-9._-]+@[a-zA-Z0-9-]+\.[a-zA-Z.]{2,5}$/]"`

- ✚ `"/.../"` đây là bắt đầu và kết thúc biểu thức chính quy
- ✚ `"^[a-zA-Z0-9._-]"` : khớp với bất kỳ chữ in thường hoặc in hoa, số từ 0 đến 9 và dấu chấm, dấu gạch dưới hoặc dấu gạch ngang
- ✚ `"+[a-zA-Z0-9-]"` : khớp với biểu tượng `@` theo sau là chữ thường hoặc chữ in hoa (a - z và A-Z), các số từ 0 đến 9 hoặc dấu gạch ngang.
- ✚ `"+\.[a-zA-Z.]{2,5}$/"` : tiếp đến là dấu chấm và khớp với bất kỳ chuỗi nào là chữ thường hoặc chữ in hoa (a - z và A-Z), các số từ 0 đến 9 với độ dài từ 2 đến 5 ký tự. Kết thúc chuỗi bằng cách thêm ký tự `$` vào sau cùng.



Chương 5.3

CẤU TRÚC ĐIỀU KHIỂN VÀ VÒNG LẶP

ThS. Thiều Thanh Quang phú

Chương 5.3. Cấu trúc điều khiển và vòng lặp

3.1. Cấu trúc điều khiển

- ✚ if ... else ...

- ✚ switch ... case ...

3.2. Vòng lặp

- ✚ for

- ✚ while

- ✚ do ... while

- ✚ Foreach

3.1. Cấu trúc điều khiển

❖ Lệnh if

✚ Ví dụ:

```
1  $so_can_kiem_tra = 12;  
2  $so_du = $so_can_kiem_tra % 2;  
3  if ($so_du == 0){  
4      echo 'Số '.$so_can_kiem_tra.' Là Số Chẵn';  
5  }
```

3.1. Cấu trúc điều khiển

❖ Lệnh if else

✚ Ví dụ:

```
1  $nam = 2014;  
2  $so_du = $nam % 2;  
3  if ($so_du == 0){  
4      echo 'Năm ' . $nam . ' Là Năm Chẵn';  
5  }  
6  else{  
7      echo 'Năm ' . $nam . ' Là Năm Lẻ';  
8  }
```

3.1. Cấu trúc điều khiển

❖ Kết hợp nhiều lệnh `if else`

✚ Ví dụ:

```
1  $mau = 'màu xanh';
2
3  if ($mau == 'màu xanh'){
4      echo 'Đây là màu xanh';
5  }
6  else if ($mau == 'màu đỏ')
7  {
8      echo "Đây là màu đỏ";
9  }
10 else if ($mau == 'màu vàng'){
11     echo 'Đây là màu vàng';
12 }
13 else{
14     echo 'Các màu khác';
15 }
```

3.1. Cấu trúc điều khiển

❖ Lệnh switch case

✚ Ví dụ:

```
1  switch ($variable) {  
2      case $value_1:  
3          // chuỗi câu lệnh  
4          break;  
5      case $value_2:  
6          // chuỗi câu lệnh  
7          break;  
8      default:  
9          // chuỗi câu lệnh  
10         break;  
11 }
```

```
1  $number = 1;  
2  switch ($number)  
3  {  
4      case 0 :  
5          echo 'Số không';  
6          break;  
7      case 1:  
8          echo 'Số một';  
9          break;  
10     case 2:  
11         echo 'Số hai';  
12         break;  
13     case 3:  
14         echo 'Số ba';  
15         break;  
16     case 4 :  
17         echo 'Số bốn';  
18         break;  
19     default:  
20         echo 'Không tìm thấy';  
21         break;  
22 }
```

3.2. Vòng lặp

❖ Vòng lặp for

✚ Ví dụ:

```
1 $sinhvien = array(  
2 'Nguyễn A',  
3 'Nguyễn B',  
4 'Nguyễn C',  
5 'Nguyễn D',  
6 'Nguyễn E',  
7 'Nguyễn F'  
8 );
```

```
1 $count = count($sinhvien);  
2 for ($i = 0; $i < $count; $i++){  
3     echo $sinhvien[$i];  
4 }
```

3.2. Vòng lặp

❖ Vòng lặp foreach

✚ Ví dụ:

```
1 // Danh sách mã số sinh viên và sinh viên tương ứng
2 $sinhvien = array(
3     'SV001' => 'Nguyễn Văn A',
4     'SV002' => 'Nguyễn Văn B',
5     'SV003' => 'Nguyễn Văn C',
6     'SV004' => 'Nguyễn Văn D',
7     'SV005' => 'Nguyễn Văn E'
8 );
9
10 // Xuất ra danh sách sinh viên
11 foreach ($sinhvien as $tensv){
12     echo $tensv . '<br/>';
13 }
```

3.2. Vòng lặp

❖ Vòng lặp while và do while

✚ Ví dụ:

```
// Cho Danh Sách Năm
$nam = array( 1990, 1991, 1992, 1993, 1994, 1995 );

// Xuất theo cách thông thường
echo $nam[0];    echo $nam[1];    echo $nam[2];
echo $nam[3];    echo $nam[4];    echo $nam[5];

// Dùng while
$i = 0;
while ($i <= 5) {
    echo $nam[$i];
    $i++; // Tăng biến $i
}

// Dùng do .. while
$i = 0;
do {
    echo $nam[$i];
    $i++;
}while ($i <=5);
```


A faint, light gray world map is visible in the background, centered behind the text.

Chương 5.4

ĐỐI TƯỢNG VÀ PHƯƠNG THỨC

ThS. Thiều Thanh Quang phú

Chương 5.4. Đối tượng và Phương thức

- 4.1. Đối tượng trong PHP
- 4.2. Phương thức trong PHP
- 4.3. Các phương thức kiểm tra dữ liệu
- 4.4. Các phương thức xử lý chuỗi
- 4.5. Các phương thức xử lý mảng
- 4.6. Các phương thức xử lý file
- 4.7. Các phương xử lý thức thời gian
- 4.8. Phương Header
- 4.9. Phương thức GET và POST

4.1. Đối tượng trong PHP

❖ Khai báo đối tượng

```
<?php
class Fruit { //khai báo lớp
    public $name; //khai báo thuộc tính
    public $color;
    function set_name($name) { //khai báo phương thức
        $this->name = $name;
    }
    function get_name() {
        return $this->name;
    }
}

$apple = new Fruit(); //tạo đối tượng 1
$apple->set_name('Apple'); //gọi phương thức
$banana = new Fruit();
$banana->set_name('Banana'); //tạo đối tượng 2
echo $apple->get_name();
echo "<br>";
echo $banana->get_name();
?>
```

Apple
Banana

4.1. Đối tượng trong PHP

- ❖ Từ khóa `this`:
 - ✚ Chỉ đối tượng hiện tại
 - ✚ Được sử dụng bên trong các phương thức

```
<?php
class Fruit {
    public $name;
    function set_name($name) {
        $this->name = $name;
    }
}
$banana = new Fruit();
$banana->set_name("Banana");
?>
```

Thay đổi giá trị thuộc tính ở bên trong lớp dùng `this`

```
<?php
class Fruit {
    public $name;
}
$banana = new Fruit();
$banana->name = "Banana";
?>
```

Thay đổi giá trị thuộc tính ở bên ngoài lớp

4.1. Đối tượng trong PHP

❖ Phương thức khởi tạo __construct()

- ✚ Khi tạo một đối tượng từ lớp thì hàm __construct() sẽ tự động được gọi đến.

Model: Nokia 8.1
Màu sắc: Black
Giá tiền: 5.000.000

```
<?php
class Mobile{
    public $model;
    public $color;
    public $price;
    function __construct($input_model, $input_color, $input_price){
        $this->model = $input_model;
        $this->color = $input_color;
        $this->price = $input_price;
    }
    function get_model(){
        return $this->model;
    }
    function get_color(){
        return $this->color;
    }
    function get_price(){
        return $this->price;
    }
}

$nokia = new Mobile('Nokia 8.1','Black','5.000.000');
echo "Model: " . $nokia->model . "<br>";
echo "Màu sắc: " . $nokia->color . "<br>";
echo "Giá tiền: " . $nokia->price;
```

4.1. Đối tượng trong PHP

- ❖ Phương thức `__destruct()`
 - ✚ Thường được khai báo bên trong một lớp
 - ✚ PHP sẽ tự động gọi hàm này ở cuối tập lệnh.

----- Dưới đây là thông tin sản phẩm -----
Model Samsung Galaxy 3
Màu sắc: Blue
Giá sản phẩm: 3.500.000

```
<?php
class Mobile{
    public $model;
    public $color;
    public $price;
    function __construct($input_model, $input_color, $input_price){
        $this->model = $input_model;
        $this->color = $input_color;
        $this->price = $input_price;
    }
    function __destruct(){
        echo "Model {$this->model} <br> Màu sắc: {$this->color}
        <br> Giá sản phẩm: {$this->price}";
    }
}

$samsung = new Mobile("Samsung Galaxy 3", "Blue", "3.500.000");
echo "----- Dưới đây là thông tin sản phẩm ----- <br>";

?>
```

4.1. Đối tượng trong PHP

❖ Kế thừa `extends`

- ✚ Lớp được dẫn xuất từ lớp khác thì được gọi là “lớp con”, nó sẽ thừa hưởng tất cả các thuộc tính và phương thức (thuộc loại public & protected) của lớp cha
- ✚ Các phương thức được kế thừa từ lớp cha có thể được ghi đè bằng cách khai báo lại bên trong lớp con.

Tôi tên là Nguyễn Văn A, sinh năm 2003, giới tính Nam

```
<?php
class CongDan{
    public $name;
    public $year;
    public function __construct($input_name, $input_year){
        $this->name = $input_name;
        $this->year = $input_year;
    }
    protected function intro(){
        echo "Tôi tên là {$this->name}, sinh năm {$this->year}";
    }
}

class SinhVien extends CongDan{ //lớp SinhVien kế thừa lớp CongDan
    public $gender;
    public function __construct($input_name, $input_year, $input_gender){
        $this->name = $input_name;
        $this->year = $input_year;
        $this->gender = $input_gender;
    }
    public function intro() { //ghi đè phương thức cùng tên
        //với phương thức của lớp cha
        echo "Tôi tên là {$this->name}, sinh năm {$this->year},
        giới tính {$this->gender}";
    }
}

$nhan = new SinhVien("Nguyễn Văn A", 2003, "Nam");
$nhan->intro();

?>
```

4.1. Đối tượng trong PHP

❖ Từ khóa `final`

- ✚ Ngăn chặn việc kế thừa lớp, hoặc ngăn chặn việc ghi đè lên phương thức.

```
<?php
final class CongDan{
    public $name;
    public function __construct($input_name){
        $this->name = $input_name;
    }
    public function intro(){
        echo "Tôi tên là {$this->name}";
    }
}
class SinhVien extends CongDan{ //lỗi
//do lớp cha không cho kế thừa
    //some code
}
?>
```

```
<?php
class CongDan{
    public $name;
    public function __construct($input_name){
        $this->name = $input_name;
    }
    //sử dụng từ khóa final để ngăn chặn ghi
    //đề phương thức
    final public function intro(){
        echo "Tôi tên là {$this->name}";
    }
}
class SinhVien extends CongDan{
    public function intro(){ //lỗi do lớp cha
        //không cho ghi đè
        echo "Chào các bạn, tôi tên là {$this->name}";
    }
}
?>
```


4.1. Đối tượng trong PHP

❖ Lớp trừu tượng **abstract**

- ✚ Lớp này phải có chứa ít nhất một “phương thức trừu tượng”.
- ✚ “Phương thức trừu tượng” là phương thức chỉ được khai báo tên, không chứa hành vi, nó cần phải được khai báo lại (bổ sung hành vi) bên trong lớp con để hoàn tất việc khai báo.
- ✚ Một lớp hoặc phương thức trừu tượng sẽ được khai báo bằng từ khóa **abstract**.

```
<?php
abstract class CongDan{
    public $name;
    public function __construct($input_name){
        $this->name = $input_name;
    }
    abstract public function intro();
}
class SinhVien extends CongDan{
    public function intro(){
        echo "<p>Chào các bạn, mình tên là {$this->name}</p>";
    }
}
class GiangVien extends CongDan{
    public function intro(){
        echo "<p>Chào các em, cô tên là {$this->name}</p>";
    }
}
$sv = new SinhVien("Minh");
$sv->intro();
$gv = new GiangVien("Lan");
$gv->intro();
?>
```

Chào các bạn, mình tên là Minh

Chào các em, cô tên là Lan

4.1. Đối tượng trong PHP

❖ Lớp giao diện **Interface**

- ✚ Chỉ định những phương thức mà một lớp nên triển khai
- ✚ Các Interface sẽ được khai báo bằng từ khóa **interface**.
- ✚ Để triển khai một Interface cho lớp thì ta sử dụng từ khóa **implements**.

meo meo
gâu gâu
chít chít

```
<?php
interface Animal{
    function makeSound();
}
class Cat implements Animal{
    public function makeSound(){
        echo "meo meo";
    }
}
class Dog implements Animal{
    public function makeSound(){
        echo "gâu gâu";
    }
}
class Mouse implements Animal{
    public function makeSound(){
        echo "chít chít";
    }
}
$cat = new Cat();
$dog = new Dog();
$mouse = new Mouse();
$animals = array($cat, $dog, $mouse);
foreach ($animals as $animal) {
    $animal->makeSound();
    echo "<br>";
}
?>
```

4.1. Đối tượng trong PHP

❖ So sánh Interface với Abstract

- ✚ Interface không được phép có các thuộc tính, trong khi Abstract có thể có.
- ✚ Các phương thức trong Interface có phạm vi truy cập mặc định là public & bắt buộc phải là public (việc sử dụng từ khóa public khi khai báo phương thức là không cần thiết), trong khi Abstract có thể là public hoặc protected.
- ✚ Tất cả các phương thức trong Interface đều mặc định là trừu tượng, không dùng từ khóa abstract khi khai báo, cũng không thể viết code cho nó.
- ✚ Các lớp có thể kế thừa từ một lớp khác, đồng thời có thể triển khai một Interface.

4.1. Đối tượng trong PHP

❖ Phương thức tĩnh Static

- ✚ Là phương thức có thể được gọi trực tiếp mà không cần phải tạo một đối tượng từ lớp chứa phương thức đó.
- ✚ Phương thức tĩnh được khai báo bởi từ khóa static
- ✚ Khi muốn truy cập một phương thức tĩnh thì chúng ta sử dụng tên lớp, theo sau là cặp dấu hai chấm :: và tên phương thức tĩnh.

```
ClassName::staticMethod();
```

```
<?php
class Greeting{
    public static function welcome(){
        echo "Chào cả nhà";
    }
}
//phương thức tĩnh được gọi thông qua tên lớp
Greeting::welcome();
?>
```

```
<?php
class Greeting{
    public static function welcome(){
        echo "Chào cả nhà";
    }
}
class Hello{
    public function __construct(){
        //phương thức tĩnh được truy cập từ lớp khác
        Greeting::welcome();
    }
}
$a = new Hello();
?>
```

4.2. Phương thức trong PHP

❖ Khai báo phương thức với từ khóa **function**

fun_name: tên phương thức

\$vars: danh sách tham số

return: giá trị trả về nếu có

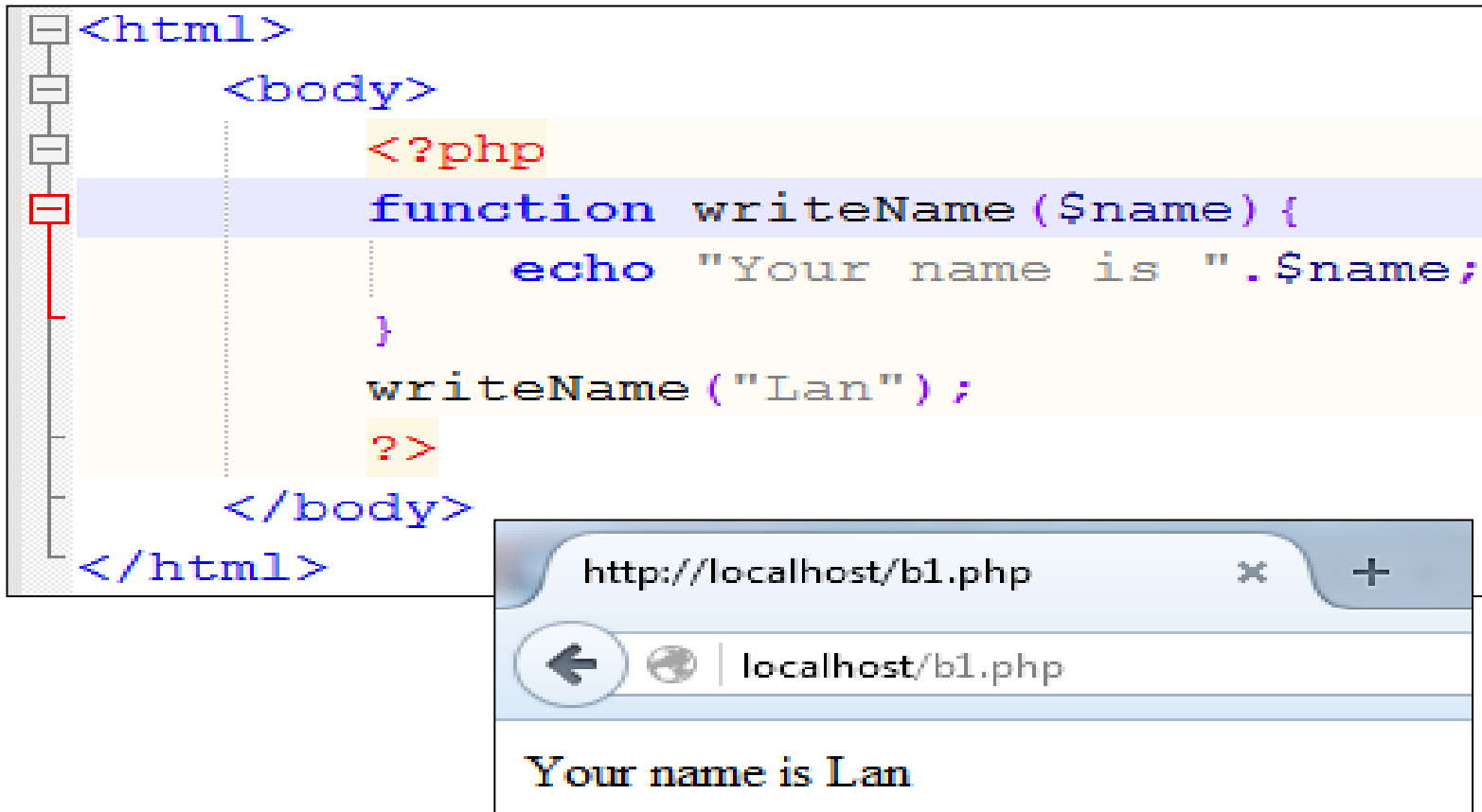
❖ Ví dụ:

```
$number = 12;  
kiem_tra_so_chan($number);  
function kiem_tra_so_chan($number) {  
    if ($number % 2 == 0) {  
        echo 'Số chẵn';  
    }  
    else {  
        echo 'Số lẻ';  
    }  
}
```

```
1 function func_name($vars)  
2 {  
3     // các đoạn code  
4     return $val;  
5 }
```

4.2. Phương thức trong PHP

❖ Phương thức có tham số



The image shows a code editor on the left and a web browser on the right. The code editor displays an HTML document with a PHP function defined and called. The function, `writeName`, takes a parameter `$name` and echoes a string that includes the parameter's value. The browser window shows the output of this function, which is "Your name is Lan".

```
<html>
  <body>
    <?php
      function writeName($name) {
        echo "Your name is ".$name;
      }
      writeName("Lan");
    ?>
  </body>
</html>
```

http://localhost/b1.php x +

localhost/b1.php

Your name is Lan

4.2. Phương thức trong PHP

❖ Phương thức trả về giá trị

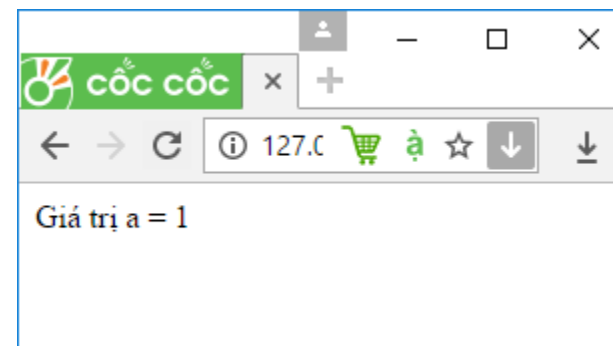
```
<html>
<body>
<?php
function add($x,$y) {
    $total = $x + $y;
    return $total;
}
echo "1 + 16 = " . add(1,16);
?>
</body>
</html>
```



4.2. Phương thức trong PHP

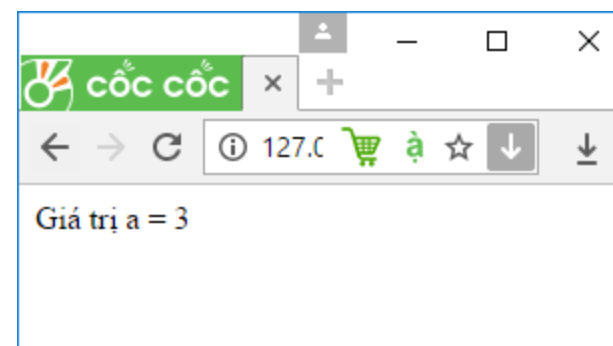
❖ Truyền bằng giá trị:

```
$a = 1;  
function tang($a) {  
    $a = $a + 2;  
    return $a;  
}  
tang($a);  
echo "Giá trị a = " . $a;
```



❖ Truyền bằng tham chiếu:

```
$a = 1;  
function tang(&$a) {  
    $a = $a + 2;  
    return $a;  
}  
tang($a);  
echo "Giá trị a = " . $a;
```



4.2. Phương thức trong PHP

❖ Các phương thức sử dụng lại file trong PHP

- # require(“đường dẫn đến tập tin PHP”)
- # include(“đường dẫn đến tập tin PHP”)
- # require_once(“đường dẫn đến tập tin PHP”)
- # include_once(“đường dẫn đến tập tin PHP”)

4.2. Phương thức trong PHP

❖ Include

- ✚ Chèn nội dung của một file khác vào file PHP hiện tại
- ✚ Tiết kiệm thời gian, nhất quán, giúp cho việc code ứng dụng nhanh hơn.

✚ Cú pháp:

```
include 'filename';
```

✚ B1: Tạo file *show.php* như sau:

```
<?php  
    echo "Xin chào ";  
    $ten = "Nam";  
?>
```

4.2. Phương thức trong PHP

❖ Include

- # **B2:** Tạo file *bai2.php* và chèn nội dung file *show.php* vào *bai2.php* bằng hàm `include()`

```
<?php
    include 'show.php';
    echo $ten;
?>
```

- # **B3:** Mở trình duyệt, gõ <http://localhost/bai2.php>

Xin chào Nam

4.2. Phương thức trong PHP

❖ Tương tự cho phương thức

✚ include_once

✚ require

✚ require_once

4.2. Phương thức trong PHP

❖ Phân biệt include và require

✚ Khi có lỗi xảy ra (file được chèn không tồn tại):

- include() sẽ hiển thị ra một thông báo lỗi dạng warning và đoạn mã vẫn tiếp tục được thực thi.
- require() sẽ hiển thị ra một thông báo lỗi dạng fatal và đoạn mã sẽ bị dừng xử lý ngay sau đó.

❖ Phân biệt include_once và require_once

✚ Là hai dạng biến đổi của hàm require() và include().

✚ Mục đích: Nếu file đã được chèn ở trước đó rồi thì sẽ không chèn nữa.

4.3. Các phương thức kiểm tra dữ liệu

❖isset()

- ✚ Kiểm tra một biến đã được khởi tạo chưa.
- ✚ Nếu đã tồn tại thì sẽ trả về TRUE, ngược lại trả về FALSE.

```
$gt = 1;  
if (isset($gt)){  
    echo 'Biến gt đã tồn tại';  
}  
else{  
    echo 'Biến gt chưa tồn tại';  
}
```

4.3. Các phương thức kiểm tra dữ liệu

❖ empty()

- ✚ Kiểm tra một biến có giá trị rỗng hoặc chưa được khởi tạo hay không.
- ✚ Giá trị của \$var là rỗng nếu nó nằm một trong các trường hợp sau:
 - \$var = 0 hoặc \$var = '0'
 - \$var = NULL
 - \$var = '';
 - \$var = FALSE
 - \$var không tồn tại

4.3. Các phương thức kiểm tra dữ liệu

❖ is_string(\$variable)

- ✚ Kiểm tra xem biến có phải là chuỗi không.
- ✚ Nếu là chuỗi thì trả về TRUE, ngược lại nếu không phải chuỗi thì trả về FALSE.

```
<?php
    $data = [];
    if(is_string($data))
        echo 'Biến này là chuỗi';
    else
        echo 'Biến này không phải là chuỗi';
    //Kết quả: Biến này không phải là chuỗi
?>
```


4.3. Các phương thức kiểm tra dữ liệu

❖ is_numeric(\$variable)

- ✚ Kiểm tra xem một biến có phải là số hay không
- ✚ Trả về TRUE nếu là số và ngược lại FALSE nếu không phải là số.

```
<?php
    $data = 1555.5;
    if(is_numeric($data))
        echo 'Biến này là số';
    else
        echo 'Biến này không phải là số';
    //Kết quả: Biến này là số
?>
```

4.3. Các phương thức kiểm tra dữ liệu

❖ is_int(\$variable)

- ✚ Kiểm tra xem một biến có phải số nguyên hay không
- ✚ Trả về TRUE nếu là số nguyên và FALSE nếu không phải.

```
<?php
    $data = 1555.5;
    if(is_int($data))
        echo 'Biến này là số nguyên';
    else
        echo 'Biến này không phải là số nguyên';
    //Kết quả: Biến này không phải là số nguyên
?>
```

4.3. Các phương thức kiểm tra dữ liệu

❖ `is_float($variable)`

- ✚ Kiểm tra xem biến có phải kiểu float hay không.
- ✚ Trả về true nếu đúng và ngược lại false nếu không phải

❖ `is_double($variable)`

- ✚ Kiểm tra xem biến có phải kiểu double hay không.
- ✚ Trả về true nếu đúng và ngược lại false nếu không phải.

4.4. Các phương thức xử lý chuỗi

❖ Trong PHP chuỗi được xác định

- ✚ Dấu nháy đơn: VD: 'abc';

- ✚ Dấu nháy kép: VD: "abc";

❖ Nếu chuỗi được đặt trong dấu nháy kép "" thì các ký tự nháy kép " bên trong chuỗi phải thêm dấu gạch '\' đằng trước nó.

```
1 | echo "Nam nói\"Cậu ấy đang ăn tối\" ";
```

❖ Nếu chuỗi được đặt trong dấu nháy kép thì trong chuỗi ta có thể truyền biến vào mà không cần dùng phép nối chuỗi.

```
1 | $str = "đang ăn tối";  
2 | echo "Nam nói\"Cậu ấy $str\" ";
```

❖ Toán tử nối chuỗi: Để nối 2 chuỗi chúng ta sử dụng toán tử (.)

- Ví dụ: echo "abc"."def";

4.4. Các phương thức xử lý chuỗi

❖ crc32 (\$str)

- ✚ Chuyển chuỗi \$str thành một dãy số nguyên.

```
<?php
    echo crc32 ( 'HTML' ) ;
?>
```

❖ ord (\$string)

- ✚ Trả về mã ASCII của ký tự đầu tiên trong chuỗi \$string.

```
1 | echo ord ( 'Ab' );
2 | // kết quả: 65
```

4.4. Các phương thức xử lý chuỗi

❖ explode (\$delimiter , \$string)

✚ Chuyển một chuỗi **\$string** thành một mảng các phần tử với ký tự tách mảng là **\$delimiter**.

```
$chuoi1 = 'Hoc lap trinh';  
//mỗi khoảng trắng là 1 phần tử trong mảng  
var_dump(explode(' ', $chuoi1));  
/*Kết quả  
array  
    0 => 'Hoc',  
    1 => 'Lap',  
    2 => 'Trinh'  
*/
```

4.4. Các phương thức xử lý chuỗi

❖ implode(\$delimiter, \$piecesarray)

- ✚ Ngược với explode, nó chuyển một mảng \$piecesarray thành chuỗi và mỗi phần tử cách nhau bởi chuỗi \$delimiter

```
echo implode(' ', array(
    'Hoc',
    'Lap',
    'Trinh'
));
//Kết quả Hoc Lap Trinh
```

❖ strlen(\$string)

- ✚ Đếm số ký tự của chuỗi \$string.

4.4. Các phương thức xử lý chuỗi

❖ str_word_count(\$str)

- ✚ Trả về số từ trong chuỗi \$str.

```
echo str_word_count('Hoc Lap Trinh');  
//Kết quả là 3
```

❖ str_repeat(\$str, int \$n)

- ✚ Lặp chuỗi \$str \$n lần.

```
<?php  
    $input = "Học Lập Trình + ";  
    $result = str_repeat($input, 3);  
    echo $result;  
?>
```


4.4. Các phương thức xử lý chuỗi

❖ `str_replace($chuoi_tim, $chuoi_thay_the, $chuoi_nguon)`

✚ Tìm kiếm và thay thế chuỗi.

```
$chuoi = 'Học CSS';  
$chuoi1 = str_replace('CSS', 'HTML', $chuoi);  
echo $chuoi1  
//Kết quả là Học HTML
```

4.4. Các phương thức xử lý chuỗi

❖ substr(\$string, \$start, \$length)

- ✚ Lấy một chuỗi con nằm trong chuỗi \$str bắt đầu từ ký tự thứ \$start và chiều dài \$length.
- ✚ **string**: chuỗi nhập vào để cắt ra chuỗi con
- ✚ **start**:
 - start > 0: chuỗi được trả về sẽ bắt đầu từ vị trí start.
 - start < 0: chuỗi con trả về sẽ được cắt từ vị trí start tính từ cuối chuỗi.
- ✚ **length**: độ dài của chuỗi trả về, length được tính đến cuối chuỗi
 - length > 0: độ dài của chuỗi trả về được tính từ đầu chuỗi
 - length < 0: độ dài của chuỗi trả về được tính từ cuối chuỗi

4.4. Các phương thức xử lý chuỗi

❖ substr(\$string, \$start, \$length)

```
echo substr('Hoc Lap Trinh', 0, 7);  
//Kết quả Hoc Lap
```

❖ strstr(\$string, \$ky_tu_cho_truoc): Tách một chuỗi bắt đầu từ \$ky_tu_cho_truoc cho đến hết chuỗi.

```
echo strstr('Hoc Lap Trinh', 'Lap');  
//Kết quả Lap Trinh
```

❖ strpos(\$str, \$chuoi_tim): Tìm vị trí của chuỗi \$chuoi_tim trong chuỗi \$str, kết quả trả về false nếu không tìm thấy.

```
echo strpos('Hoc Lap Trinh', 'Lap');  
//Kết quả là 4
```

4.4. Các phương thức xử lý chuỗi

❖ `strtolower($str);`

✚ Chuyển tất cả các ký tự chuỗi `$str` sang chữ thường

❖ `strtoupper($string);`

✚ Chuyển tất cả các ký tự chuỗi `$str` sang chữ hoa

❖ `ucwords($string)`

✚ Chuyển từ đầu tiên trong chuỗi `$string` sang chữ hoa

4.4. Các phương thức xử lý chuỗi

❖ trim(\$string, \$ky_tu)

- ✚ Xóa ký tự **\$ky_tu** nằm ở đầu và cuối chuỗi \$str, nếu ta không nhập \$ky_tu thì mặc định nó hiểu là xóa khoảng trắng.

```
echo trim('    Hoc    Lap Trinh ');  
//Kết quả Hoc Lap Trinh
```

❖ ltrim(\$string, \$ky_tu): Tương tự như trim nhưng chỉ xóa bên trái

❖ rtrim(\$string, \$ky_tu): Tương tự như trim nhưng chỉ xóa bên phải

❖ nl2br(\$string): Chuyển các ký tự xuống dòng “\n” thành thẻ

4.5. Các phương thức xử lý mảng

❖ count(\$array)

✚ Đếm xem trong mảng có bao nhiêu phần tử.

✚ VD:

```
<?php
    $array=[ 'CSS', 'PHP' ];
    echo count($array);
    // Kết quả: 2
?>
```

4.5. Các phương thức xử lý mảng

❖ array_values(\$array)

✚ Đưa mảng về dạng mảng tuần tự

✚ VD:

```
<?php
    $array=[ 'm1'=>'CSS', 'M2'=>'PHP' ];
    print_r(array_values($array));
    // Kết quả: Array ( [0] => CSS [1] => PHP )
?>
```

4.5. Các phương thức xử lý mảng

❖ array_keys(\$array)

✚ Trả về một mảng tuần tự với phần tử là key của mảng ban đầu.

✚ VD:

```
<?php
    $array=[ 'm1'=>'CSS', 'M2'=>'PHP' ];
    print_r(array_keys($array));
    // Kết quả: Array ([0] => m1 [1] => m2)
?>
```


4.5. Các phương thức xử lý mảng

❖ `array_pop($array)`

- ✚ Trả về phần tử cuối cùng của mảng.

❖ `array_push($array,$var,$var...)`

- ✚ Thêm một hoặc nhiều phần tử vào cuối mảng và trả về số lượng phần tử của mảng sau khi thêm.

❖ `array_shift($array)`

- ✚ xóa phần tử đầu tiên của mảng và trả về phần tử vừa xóa.

❖ `array_unshift($array,$var,$var...)`

- ✚ Thêm một hoặc nhiều phần tử vào đầu mảng, và trả về số lượng phần tử của mảng sau khi thêm

4.5. Các phương thức xử lý mảng

❖ `array_flip($array)`

- ✚ Chuyển đổi key của mảng thành value và ngược lại.

❖ `sort($array)`

- ✚ Sắp xếp lại mảng theo chiều tăng dần và trả về giá trị TRUE nếu thành công và ngược lại FALSE nếu không thành công.

❖ `array_reverse($array)`

- ✚ Đảo ngược lại vị trí của các phần tử trong mảng

❖ `array_merge($array,$array...)`

- ✚ Gộp hai hay nhiều mảng thành một mảng.

❖ `array_rand($array,$number)`

- ✚ Lấy ra key ngẫu nhiên trong mảng với number là số lượng muốn lấy.

4.5. Các phương thức xử lý mảng

❖ `array_search($keyword,$array)`

- ✚ Tìm kiếm giá trị của mảng và trả về key của phần tử đó nếu có.

❖ `array_slice($array,$begin,$lenght)`

- ✚ Lấy ra \$lenght phần tử bắt đầu từ \$begin trong mảng.

❖ `array_unique($array)`

- ✚ Loại bỏ các phần tử trùng nhau trong mảng và trả về một mảng mới sau khi đã loại bỏ.

❖ `array_key_exists($key,$array)`

- ✚ Kiểm tra xem mảng \$array có tồn tại khóa \$key không. Trả về TRUE nếu tồn tại và ngược lại.

4.5. Các phương thức xử lý mảng

❖ `in_array($value,$array)`

- ✚ Kiểm tra xem mảng `$array` có tồn tại giá trị `$value` không? và trả về `TRUE` nếu có và ngược lại.

❖ `array_intersect($array1,$array2,...)`

- ✚ Trả về mảng các phần tử giống nhau về `$value` giữa các mảng

❖ `array_intersect_assoc($array1,$array2,...)`

- ✚ Trả về mảng chứa các phần tử giống nhau cả `key` và `value`

❖ `is_array($array)`

- ✚ Kiểm tra xem một biến có phải mảng hay không. Trả về `true` nếu là mảng và ngược lại.

4.6. Các phương thức xử lý file

❖ Để mở file trong PHP chúng ta dùng hàm **fopen** với cú pháp:

fopen(command, mode)

+ **command**: là đường dẫn đến file các bạn muốn mở.

+ **mode** : là quyền truy cập vào file.

```
<?php
// dùng @ để ngăn chặn thông báo lỗi khi sai đường dẫn file
$file=@fopen('data.txt', 'r');
if(!$file)
    echo "Mở file không thành công";
else
    echo "Mở file thành công";
?>
```

4.6. Các phương thức xử lý file

❖ Các quyền trên file

| Mode | Diễn giải |
|------|--|
| r | Read only |
| r+ | Read + Write |
| w | Write only |
| w+ | Write + Read. Nếu file này tồn tại thì nội dung cũ sẽ bị xóa đi và ghi lại nội dung mới, còn nếu file chưa tồn tại thì nó tạo file mới |
| a | Mở dưới dạng append dữ liệu, chỉ có write và nếu file tồn tại nó sẽ ghi tiếp nội dung phía dưới, ngược lại nếu file không tồn tại nó tạo file mới |
| a+ | Mở dưới dạng append dữ liệu, bao gồm write và read. Nếu file tồn tại nó sẽ ghi tiếp nội dung phía dưới, ngược lại nếu file không tồn tại nó tạo file mới |
| b | Mở dưới dạng chế độ binary |

4.6. Các phương thức xử lý file

❖ Đọc file từng ký tự

```
$fp = @fopen('demo.txt', "r");  
// Kiểm tra file mở thành công không  
if (!$fp) {  
    echo 'Mở file không thành công';  
}  
else{  
    // Lặp qua từng ký tự để đọc  
    while(!feof($fp)) {  
        echo fgetc($fp);  
    }  
}
```

4.6. Các phương thức xử lý file

❖ Đọc file từng dòng

```
$fp = @fopen('demo.txt', "r");  
// Kiểm tra file mở thành công không  
if (!$fp) {  
    echo 'Mở file không thành công';  
}  
else{  
    // Lặp qua từng dòng để đọc  
    while(!feof($fp)){  
        echo fgets($fp);  
    }  
}
```


4.6. Các phương thức xử lý file

❖ Đọc file hết file

```
$fp = @fopen('demo.txt', "r");  
// Kiểm tra file mở thành công không  
if (!$fp) {  
    echo 'Mở file không thành công';  
}  
else{  
    // Đọc file và trả về nội dung  
    $data = fread($fp, filesize('demo.txt'));  
    echo $data;  
}
```

4.6. Các phương thức xử lý file

❖ Ghi file

```
$fp = @fopen('demo.txt', "w");  
// Kiểm tra file mở thành công không  
if (!$fp) {  
    echo 'Mở file không thành công';  
}  
else  
{  
    $data = 'freetuts.net file functions tutorial';  
    fwrite($fp, $data);  
}
```

4.6. Các phương thức xử lý file

❖ Đóng file

```
$fp = @fopen('demo.txt','w');  
if(!$fp) {  
    echo 'Mở file không thành công';  
}  
else  
    echo 'Mở file thành công';  
    fclose($fp);
```

4.6. Các phương thức xử lý file

- ❖ Kiểm tra file có tồn tại không? Dùng hàm `file_exists($path)`

```
1 if (file_exists('demo.txt'))
2 {
3     echo 'File tồn tại';
4 }
```

- ❖ Để copy sang file mới ta dùng hàm `copy($source, $dest)`

```
1 if (!copy('demo2.txt', 'demo3.txt'))
2 {
3     echo 'Copy thất bại';
4 }
```

- ❖ Dùng hàm `unlink($path)` để xóa file

```
1 if (file_exists('demo.txt'))
2 {
3     unlink('demo.txt');
4 }
```

4.6. Các phương thức xử lý file

❖ Lấy nội dung của file

- ✚ Để lấy nội dung của một file chúng ta cũng có thể dùng phương thức `file_get_contents()`

```
<?php
    echo file_get_contents('data.txt');
?>
```

❖ Đổi tên file

- ✚ Để đổi tên file dùng phương thức `rename(old,new)`

```
<?php
    rename('data.txt','newdata.txt');
?>
```

4.6. Các phương thức xử lý file

❖ Tạo thư mục

```
<?php
    mkdir(path,mode,recursive,context);
?>
```

❖ Kiểm tra thư mục

```
<?php
    if(is_dir('public'))
        echo 'Tồn tại';
    else
        echo 'Không tồn tại';
?>
```

4.6. Các phương thức xử lý file

❖ Upload file:

`$_FILES['nameInputFile']['properties']`

- ✚ `nameInputFile`: là tên của input file.
- ✚ `properties`: Bao gồm 5 thuộc tính sau đây:
 - **`name`**: Tên của file vừa upload lên.
 - **`type`**: Kiểu dữ liệu của file
 - **`tmp_name`**: Đường dẫn tạm của file ở trên server.
 - **`error`**: Trạng thái của file. 0 là không có lỗi
 - **`size`**: Dung lượng của file ở đây đơn vị là bytes.

4.6. Các phương thức xử lý file

❖ Upload hoàn tất thì dùng phương thức

`move_uploaded_file(filename, destination)`

✚ **filename**: là đường dẫn đến file tạm thời ở trên server

✚ **destination**: là đường dẫn chứa file được lưu trữ.

4.6. Các phương thức xử lý file

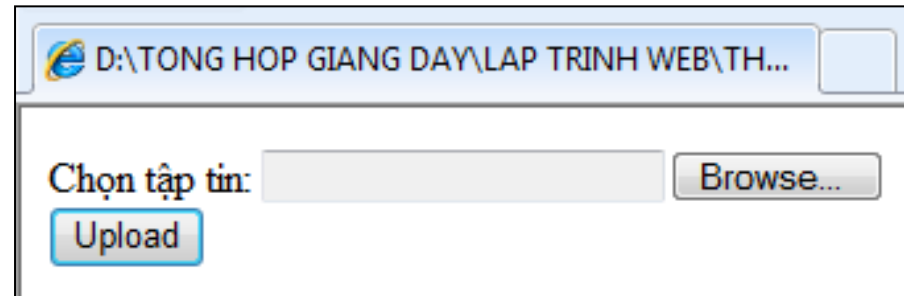
- ❖ Đối tượng **\$_FILES** chứa các thông tin liên quan đến việc upload tập tin lên server

| PHẦN TỬ | CHỨC NĂNG |
|---|--|
| <code>\$_FILES['file_upload']['name']</code> | Lấy tên file |
| <code>\$_FILES['file_upload']['type']</code> | Kiểu của file được lưu ở dạng MINE (Ví dụ: image/gif, audio/wav) |
| <code>\$_FILES['file_upload']['size']</code> | Dung lượng của file tính theo byte |
| <code>\$_FILES['file_upload']['tmp_name']</code> | lấy tên file tạm (Sau khi upload, server sẽ lưu file vào một file tạm trên server) |
| <code>move_uploaded_file (string \$filename , string \$destination)</code> | Ghi file lên server |
| <code>\$_FILES['file_upload']['error']</code> <ul style="list-style-type: none">▪ <code>UPLOAD_ERR_OK (= 0)</code>▪ <code>UPLOAD_ERR_INI_SIZE (= 1)</code>▪ <code>UPLOAD_ERR_FORM_SIZE (= 2)</code>▪ <code>UPLOAD_ERR_PARTIAL (= 3)</code>▪ <code>UPLOAD_ERR_NO_FILE (= 4)</code> | <ul style="list-style-type: none">▪ không có lỗi, quá trình upload → thành công▪ dung lượng file upload quá giới hạn mặc định trong php.ini▪ dung lượng file upload quá giới hạn bởi MAX_FILE_SIZE▪ file chỉ được upload 1 phần (lỗi đường truyền khi upload)▪ không có file nào được upload (file ở client không tồn tại) |
| ... | ... |

4.6. Các phương thức xử lý file

❖ Tạo trang form.php

```
<form action="xulyupload.php" method="post"
    enctype="multipart/form-data">
    Chọn tập tin: <input type="file" name="taptin1"/><br/>
    <input type="submit" name="submit" value="uploadclick"/>
</form>
```



❖ Lưu ý:

- ✚ Luôn sử dụng phương thức POST.
- ✚ Luôn sử dụng `enctype="multipart/form-data"` trong thẻ FORM.

4.6. Các phương thức xử lý file

❖ Tạo trang xulyupload.php

```
if (isset($_POST['uploadclick'])) {  
    // Nếu người dùng có chọn file để upload  
    if (isset($_FILES['taptin1'])) {  
        if ($_FILES["taptin1"]["error"] > 0){  
            echo "Lỗi: " . $_FILES["taptin1"]["error"] . "<br />";  
        }  
        else {  
            echo "Tên file: " . $_FILES["taptin1"]["name"] . "<br />";  
            echo "Kiểu: " . $_FILES["taptin1"]["type"] . "<br />";  
            echo "Dung lượng: " . ($_FILES["taptin1"]["size"] / 1024) . " Kb<br />";  
            echo "File tạm: " . $_FILES["taptin1"]["tmp_name"] . "<hr />";  
            // Nếu trùng tên file, thông báo file đã tồn tại  
            if (file_exists("upload/" . $_FILES["taptin1"]["name"])){  
                echo $_FILES["taptin1"]["name"] . " đã tồn tại. ";  
            }  
            else // Di chuyển file vào thư mục upload  
            {  
                move_uploaded_file($_FILES["taptin1"]["tmp_name"], "upload/"  
                    . $_FILES["taptin1"]["name"]);  
                echo "Địa chỉ file đã upload: " . "upload/" . $_FILES["taptin1"]["name"];  
            }  
        }  
    }  
}
```

4.7. Các phương xử lý thức thời gian

❖ Thiết lập thời gian ở Việt Nam

✚ Cú pháp:

`date_default_timezone_set(timezone_identifier)`

- `timezone_identifier` là tên timezone các bạn muốn xét

✚ Phải đặt phương thức này ở đầu file để các đoạn code phía sau hoạt động được chính xác.

✚ VD: Xét timezone của Việt Nam

`date_default_timezone_set('Asia/Ho_Chi_Minh');`

4.7. Các phương xử lý thức thời gian

❖ Danh sách các format trong PHP

| Ký tự | Ý nghĩa | Ví dụ |
|-------|--------------------------------------|--------|
| Y | Năm với 4 chữ số | 2015 |
| y | Năm với 2 chữ số | 15 |
| n | Tháng với 1 hoặc 2 chữ số | 7 |
| m | Tháng với 2 chữ số | 07 |
| F | Tháng | July |
| M | Tháng với 3 chữ cái | Jul |
| j | Ngày trong tháng với 1 hoặc 2 chữ số | 5 |
| d | Ngày trong tháng với 2 chữ số | 05 |
| l | Ngày trong tuần | Monday |
| D | Ngày trong tuần với 3 chữ cái | Mon |
| g | Định dạng 12 giờ có 1 hoặc 2 chữ số | 6 |
| G | Định dạng 24 giờ có 1 hoặc 2 chữ số | 18 |
| h | Định dạng 12 giờ có 2 chữ số | 06 |
| H | Định dạng 24 giờ có 2 chữ số | 18 |
| i | Phút | 21 |
| a | am hoặc pm | am |
| A | AM hoặc PM | PM |
| s | Giây | 30 |

4.7. Các phương xử lý thức thời gian

❖ Định dạng ngày tháng với hàm date()

- ✚ Dùng để chuyển đổi thời gian thành các định dạng tùy chỉnh.
- ✚ Cú pháp: `date(format,timestamp);`
 - `format` là định dạng thời gian.
 - `timestamp` là thời gian truyền vào(int) nếu để trống trường này thì PHP sẽ tự động lấy thời gian hiện tại.

4.7. Các phương xử lý thời gian

- ❖ Lấy ra ngày tháng năm hiện tại theo định dạng ngày-tháng-năm giờ:phút:giây

```
<?php
    echo date('d-m-Y H:i:s');
    //Kết quả: 04-01-2018 22:08:08
?>
```

- ❖ Truyền chuỗi trong hàm date

- ✚ Để cho nội dung chuỗi hiển thị ta thêm \ vào trước các ký tự đặc biệt trong chuỗi.

```
<?php
    echo date('\B\â\y \g\i\ờ \l\à d-m-Y H:i:s');
    //Kết quả: Bây giờ là 04-01-2018 22:11:46
?>
```

4.7. Các phương xử lý thức thời gian

❖ Ví dụ hàm date() lấy ra ngày / tháng / năm hiện tại

```
echo date("Y/m/d"), '<br>'; //kết quả: 2015/07/22
echo date("Y:m:d"), '<br>'; //kết quả: 2015:07:22
echo date("Y-m-d"), '<br>'; //kết quả: 2015-07-22
echo date("Y m d h:i:s"), '<br>'; //kết quả 2015 07 22 08:30:30
```

❖ Hàm mktime(hour, minute, second, month, day, year):

✚ Tạo ngày từ ngày giờ cụ thể.

```
$time = mktime("8", "30", "00", "07", "22", "2015");
//$time là timestamp, bạn có thể sử dụng nó trong hàm date()
echo date("Y-m-d H:i:s", $time), '<br>';
//sau 15 phút
$time += 900;
echo '15 phút sau: ', date("Y-m-d H:i:s", $time), '<br>';
//3 ngày sau
echo ' 3 ngày sau: ', date("Y-m-d H:i:s", ($time+3*60*60*24));
```


4.7. Các phương xử lý thức thời gian

❖ array getdate([int \$timestamp])

✚ Trả về một array thông tin về thời gian hiện tại.

✚ Ví dụ:

```
$today = getdate();  
//in mang thoi gian  
print_r($today);  
//in ngay thang nam hien tai  
echo "</br>Ngày: ", $today["mday"], " tháng: ",  
      $today["mon"], " năm: ", $today["year"];
```

Kết quả:

```
Array (  
  [seconds] => 51  
  [minutes] => 25  
  [hours] => 23  
  [mday] => 7  
  [wday] => 4  
  [mon] => 1  
  [year] => 2016  
  [yday] => 6  
  [weekday] => Thursday  
  [month] => January  
  [0] => 1452209151  
  
Ngày: 7 tháng: 1 năm: 2016
```

4.7. Các phương xử lý thời gian

❖ bool checkdate(int \$month, int \$day, int \$year)

✚ Kiểm tra ngày, tháng, năm đưa vào có hợp lệ hay không?

```
<?php
    var_dump(checkdate(2, 29, 2015)); //False
    var_dump(checkdate(4, 18, 2014)); //True
?>
```

❖ Giới hạn của tham số:

✚ Tháng (\$month) từ 1 đến 12,

✚ Ngày (\$day) từ 1 đến 31, phụ thuộc theo tháng

✚ Năm (\$year) từ 1 đến 32767

4.7. Các phương xử lý thức thời gian

❖ int time(void)

- ✚ Trả về thời gian hiện tại được tính bằng giây kể từ 0:0:0 1/1/1970 GMT

```
<?php
    $today = time();
    echo $today;
    //kết quả: 1452209807
?>
```

❖ Chuyển đổi thời gian sang kiểu int

```
<?php
    echo strtotime(date('Y-m-d H:i:s'));
?>
```

4.8. Phương Header

❖ Cú pháp

```
1 | header ($string, $replace = true, $http_response_code = null) {}
```

- ✚ \$string: Quyết định hành động server sẽ làm gì.
- ✚ \$replace: tham số này được đặt mặc định là true. Tham số này quyết định chuỗi \$string được replace hay là khai báo mới trong trường hợp khai báo nhiều header
- ✚ \$http_response_code: Mã code trả về từ Server. VD 404 là not found 301 là chuyển hướng có chủ đích

4.8. Phương Header

❖ Điều hướng với header()

- ✚ Cú pháp: **header('location:' . \$url);**
 - **location**: khai báo chuyển trang
 - **\$url** là địa chỉ muốn chuyển hướng tới.

✚ Ví dụ:

```
1 | header('Location: http://www.domain.net/');
```

```
<?php
```

```
header('Location: http://www.domain.net/', true, 301);
```

```
//Khi thay đổi domain của website đã thay đổi
```

```
//thì khi người dùng truy cập vào domain cũ,
```

```
//ta sẽ chuyển hướng nó sang domain mới bằng cách
```

```
//sử dụng code là 301 vì đây là code khai báo chuyển hướng có điều kiện
```

```
?>
```

4.8. Phương Header

❖ Thay đổi kiểu chữ

- ✚ Khắc phục tình trạng lỗi font khi trả kết quả về không có định dạng thẻ meta utf8 bằng cách đặt đoạn code sau ở đầu file.

```
header('Content-Type: text/html; charset=utf-8');
```

❖ Khai báo định dạng dữ liệu trả về

```
header("Content-type: text/javascript");
```

- ✚ Định dạng trả về kiểu image png:

```
header("Content-type: image/png");
```

- ✚ -Định dạng trả về kiểu json:

```
header("Content-type: application/json");
```

4.9. Phương thức GET và POST

❖ Có 2 cách gửi dữ liệu từ Client lên Server đó là dùng phương thức GET hoặc phương thức POST

❖ GET

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

✚ Ví dụ:

http://www.phpvn.org/topic.php?TOPIC_ID=161

➤ Khi đó, trình duyệt sẽ gửi theo địa chỉ trên một cặp: **biến = giá trị**, trong đó biến có tên là **TOPIC_ID** và giá trị là **161** (**TOPIC_ID=161**).

4.9. Phương thức GET và POST

❖ GET

✚ Dữ liệu gửi có thể đưa lên nhiều cặp **biến = giá_trị** bằng cách phân cách chúng bởi dấu **&**

✚ Ví dụ:

```
http://www.phpvn.org/index.php?method=Reply&TOPIC_ID=161&FORUM_ID=20
```

✚ Với địa chỉ URL trên, trình duyệt gửi lên 3 cặp biến = giá_trị theo phương thức GET:

- method=Reply
- TOPIC_ID=161
- FORUM_ID=20.

4.9. Phương thức GET và POST

❖ GET

- ✚ Khi trình duyệt gửi các thông tin này lên server, PHP sẽ tự động sinh ra một mảng có tên là **\$_GET[]** để chứa tất cả các cặp biến và giá trị đó.

```
$_GET = array(  
    'method' => 'Reply',  
    'TOPIC_ID' => '161',  
    'FORUM_ID' => '20'  
);
```

- ✚ Trong đó:

- *chỉ số* = một chuỗi mang tên của tên biến
- *giá trị* của chỉ số đó = giá trị của biến do trình duyệt gửi lên.

4.9. Phương thức GET và POST

❖ GET

✚ Trang giao diện: *giaodien.php*

```
<body>
    <a href='chitietsach.php?Ma=N001'>Chi tiết </a>
</body>
```

✚ Trang giao diện: *chitietsach.php*

```
<body>
    <?php
        if(isset($_GET["Ma"])){
            if($_GET["Ma"] == "N001"){
                echo "Bạn đã chọn mua sách có mã số ". $_GET["Ma"];
            }else{
                echo "Bạn chọn mua sách khác";
            }
        }else{
            echo "Dữ liệu không hợp lệ";
        }
    ?>
</body>
```

4.9. Phương thức GET và POST

❖ POST

- ✚ Dữ liệu gửi từ trình duyệt lên server qua phương thức POST là phần dữ liệu được lưu trữ trong phần thân Request.
- ✚ POST sẽ gửi dữ liệu qua Form HTML
 - Các giá trị sẽ được định nghĩa trong các thẻ input như *textbox*, *radio*, *checkbox*, *password*, *textarea*, *hidden*.
 - Nhận dạng thông qua tên (name) của các thẻ input

4.9. Phương thức GET và POST

❖ POST

✚ Truyền theo GET và POST

→ Truyền theo phương thức GET
http://www.phpvn.org/topic.php?TOPIC_ID=161
→ Truyền theo phương thức POST
<http://www.phpvn.org/topic.php>

- ✚ Khi đó, trình duyệt cũng sẽ gửi lên server một cặp: **biến = giá trị**, trong đó biến có tên là TOPIC_ID và giá trị là 161 (TOPIC_ID=161)
- ✚ Khi trình duyệt gửi các thông tin này lên server, PHP sẽ tự động sinh ra một mảng có tên là **\$_POST[]** để chứa tất cả các cặp biến và giá trị đó.

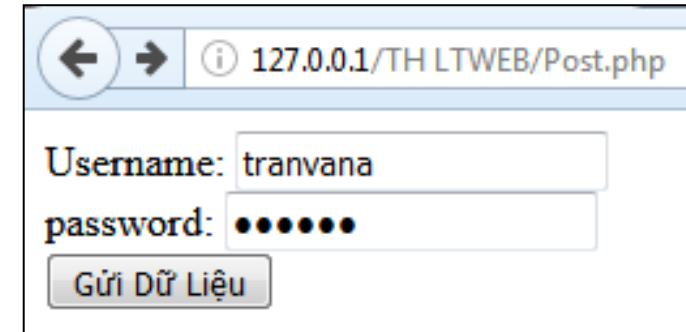
```
$_POST = array(  
    'TOPIC_ID' => '161'  
);
```

4.9. Phương thức GET và POST

❖ POST

✚ Trang form.php

```
<body>
  <form method="POST" action="xulydangnhap.php" name="form1">
    Username: <input type="text" name="username" value=""/> <br/>
    password: <input type="password" name="password" value=""/> <br/>
    <input type="submit" name="form_click" value="Gửi Dữ Liệu"/>
  </form>
</body>
```



127.0.0.1/TH LTWEB/Post.php

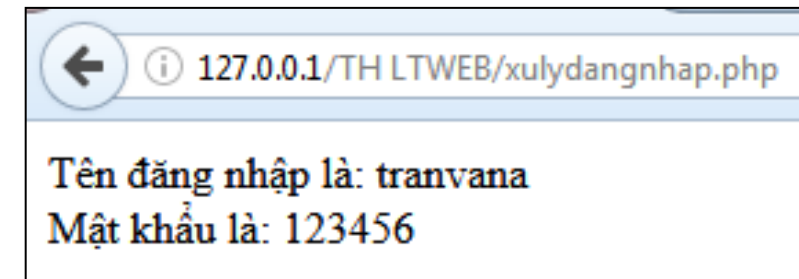
Username: tranvana

password: ••••••

Gửi Dữ Liệu

✚ Trang xulydangnhap.php

```
<body>
  <?php
    if (isset($_POST['form_click'])) {
      echo 'Tên đăng nhập là: ' . $_POST['username'];
      echo '<br/>';
      echo 'Mật khẩu là: ' . $_POST['password'];
    }
  ?>
</body>
```



127.0.0.1/TH LTWEB/xulydangnhap.php

Tên đăng nhập là: tranvana

Mật khẩu là: 123456

4.9. Phương thức GET và POST

❖ So sánh GET và POST

- ✚ POST bảo mật hơn GET vì dữ liệu được gửi ngầm
- ✚ GET luôn luôn nhanh hơn POST vì dữ liệu gửi đi được Browser giữ lại trong cache. Khi cần thì Browser trả về ngay chứ không cần gửi lên Server.
- ✚ Khi nào dùng GET – POST
 - Khi SEO thì phải sử dụng GET.
 - Khi dữ liệu bạn không cần bảo mật thì dùng GET
 - Khi đăng nhập, Comment, đăng tin dùng phương thức POST. Còn khi lấy tin ra thì dùng phương thức GET...
 - Khi sử dụng lệnh select thì dùng GET, khi sử dụng lệnh insert update, delete thì nên dùng POST.

4.9. Phương thức GET và POST

❖ REQUEST

- ✚ Chứa nội dung của các biến \$_GET, \$_POST và \$_COOKIE
- ✚ Thường dùng để lấy kết quả dữ liệu gửi từ form mà không cần quan tâm phương thức truyền dữ liệu là GET hay POST.

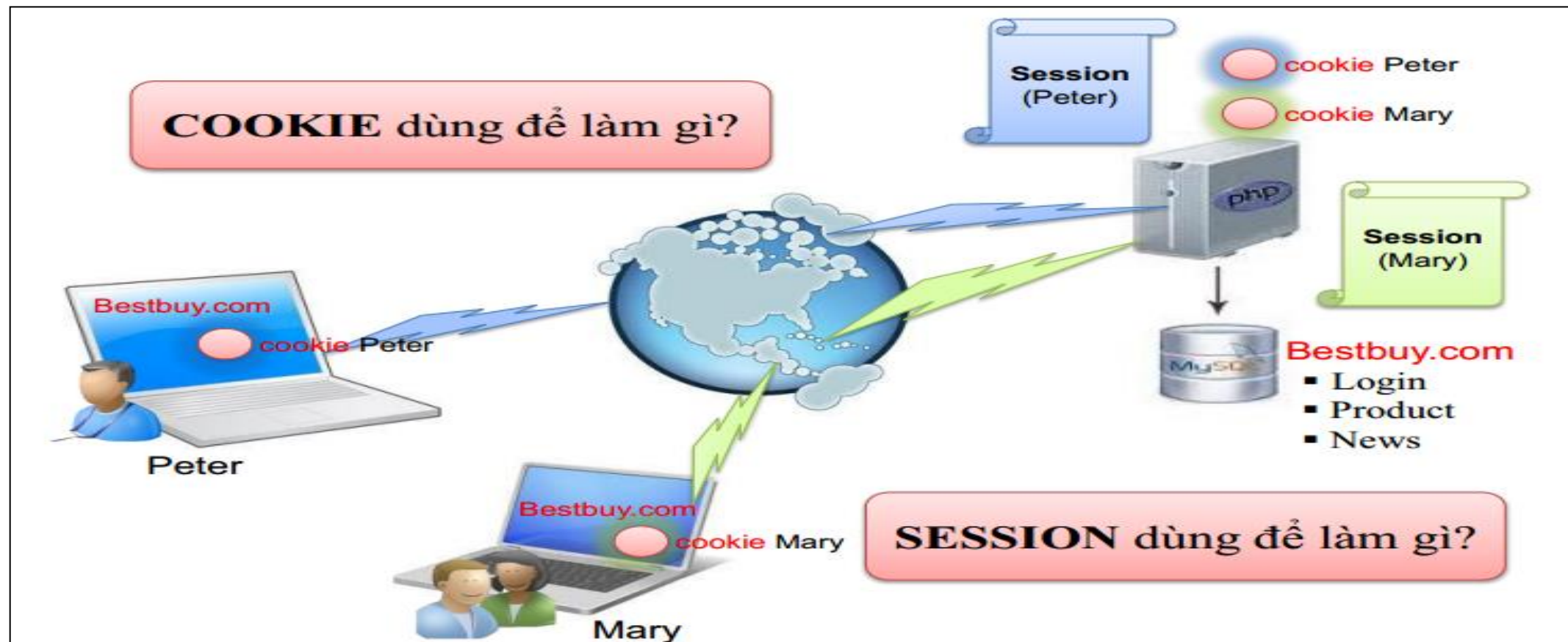


Chương 5.5

SESSION VÀ COOKIE

ThS. Thiều Thanh Quang phú

Chương 5.5. Session và cookie



➔ Sau khi đăng nhập [gmail.com](https://mail.google.com), mỗi lần đọc mail, gửi mail,... người dùng có cần đăng nhập lại?

Chương 5.5. Session và cookie

5.1. Session

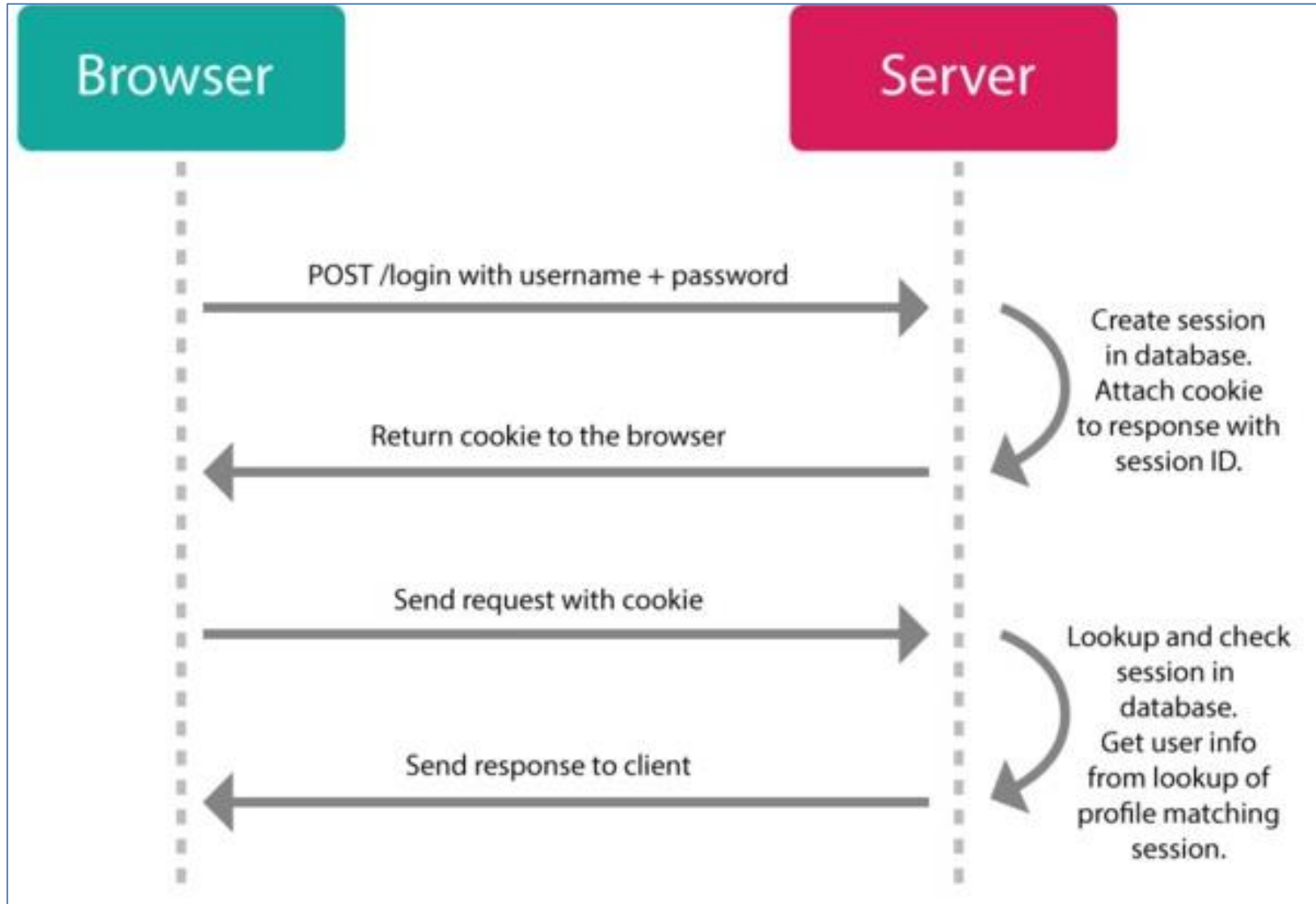
5.2. Cookie

5.1. Session

- ❖ Session (phiên truy cập): là đoạn dữ liệu được lưu trong 1 file trong một thư mục tạm thời trên Server; Dữ liệu này sẽ có sẵn cho tất cả các trang trên website trong suốt quá trình truy cập trang đó.
- ❖ Session lưu trữ thông tin: thông tin người dùng, thông tin cấu hình,...
- ❖ Session là khoảng thời gian người sử dụng giao tiếp với 1 ứng dụng. Một session được bắt đầu khi người sử dụng truy cập vào ứng dụng lần đầu tiên đến khi kết thúc phiên làm việc. Ví dụ đăng nhập đến lúc đăng xuất tài khoản mail
- ❖ Mỗi client sẽ có một SessionID khác nhau nên hoàn toàn không có thể xảy ra việc xung đột session. Ứng dụng thực tế trong việc sử dụng session thường làm chức năng đăng nhập, giỏ hàng,...



5.1. Session



5.1. Session

❖ Các phương thức

- **session_start():** khởi tạo session.
- **session_register(tên biến):** đăng ký biến session
- **\$_SESSION[tên_session] = giá_trị;** đặt giá trị cho session
- **\$_SESSION[tên_session]** đọc giá trị từ session
- **session_destroy();** hủy tất cả các dữ liệu trong session
- **session_unset();** hủy tất các biến trong session
- **session_unregister(tên biến)** hủy 1 biến trong session

5.1. Session

❖ Khởi tạo session:

✚ Để sử dụng được session cần đặt ở đầu mỗi tập tin

session_start()

❖ Đăng ký 1 giá trị cho session

session_register("user")

❖ Lưu trữ session: sử dụng biến toàn cục \$_SESSION

❖ Thêm session.

\$_SESSION['user'] = 'admin';

❖ Lấy giá trị của session:

\$_SESSION["user"]

5.1. Session

❖ Kiểm tra session có tồn tại không:

```
<?php
if ( isset($_SESSION['user']) )
{
    echo $_SESSION['user'];
}
?>
```

5.1. Session

❖ Hủy bỏ session:

✚ Để hủy bỏ giá trị của session có những cách sau:

- Cho phép hủy bỏ toàn bộ giá trị của session

`session_destroy()`

- Cho phép hủy bỏ một session

`unset($_SESSION['name']);`

```
1  <?php
2      session_start();
3      session_destroy();
4  ?>
5  <html>
6  <head>
7  <title>Test page 1</title></head>
8  <body>
9  <b><a href="session2.php">Click here</a></b>
10 </body>
11 </html>
```


5.1. Session

❖ Tạo trang *session.php* với nội dung sau:

```
1  <?php
2      session_start();
3      session_register("name");
4      $_SESSION["name"] = "Kenny Huy";
5  ?>
6  <html>
7  <head>
8  <title>Test page 1</title></head>
9  <body>
10 <b><a href="session2.php">Click here</a></b>
11 </body>
12 </html>
```

❖ Tạo trang *session2.php* với nội dung sau:

```
1  <?php
2      session_start();
3  ?>
4  <html>
5  <head><title>Result Page</title></head>
6  <body>
7  <?php
8      echo "Ten cua ban la <b>".$_SESSION["name"]."</b>";
9  ?>
10 </body>
11 </html>
```

5.1. Session

❖ Ví dụ:

✚ Có 4 trang web

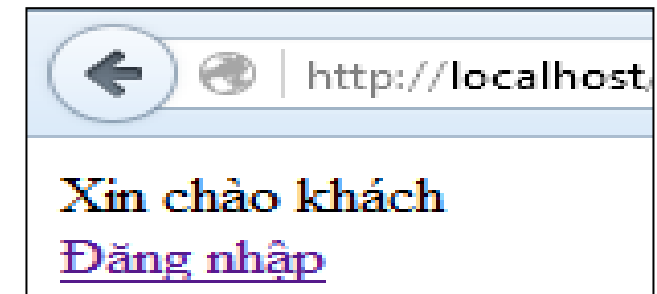
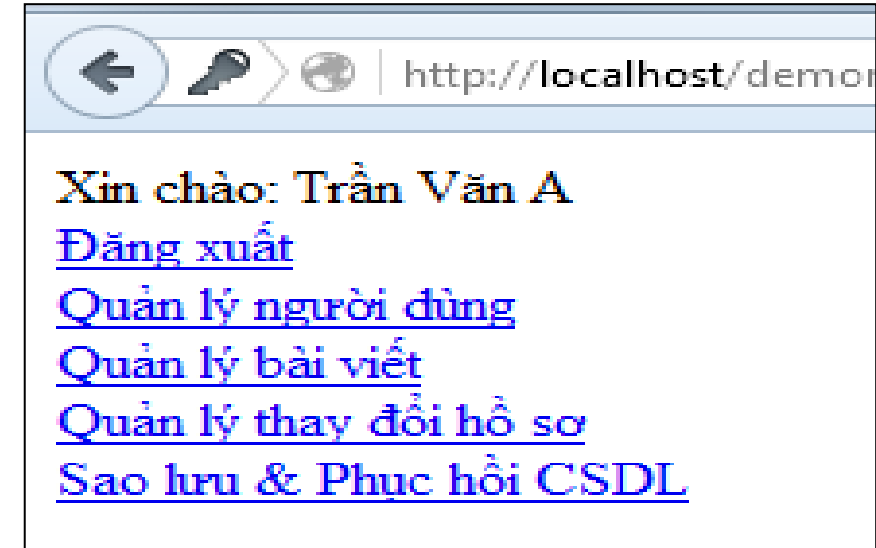
- *dangnhap.php.*
- *dangnhap_submit.php.*
- *quanly.php*
- *dangxuat.php.*

✚ Khi chạy trang web sẽ bắt đầu ở trang *dangnhap.php*

| ĐĂNG NHẬP HỆ THỐNG | |
|--|--------------------------|
| Tên đăng nhập | <input type="text"/> |
| Mật khẩu | <input type="password"/> |
| <input type="button" value="Đăng nhập"/> | |

5.1. Session

- ❖ Sau khi nhấn nút Đăng nhập, sẽ chuyển qua **dangnhap_submit.php** để xử lý
 - ✚ Nếu đăng nhập thất bại sẽ báo lỗi
 - ✚ Nếu đăng nhập thành công sẽ tiến hành đăng ký SESSION và chuyển hướng về trang **quanly.php**
- ❖ Khi nhấn liên kết Đăng xuất sẽ chuyển tới trang **dangxuat.php**, tiến hành hủy SESSION sau đó chuyển hướng về trang **quanly.php**



5.1. Session

```
<form action="dangnhap_submit.php" method="POST">
  <table border="1" align="center" cellpadding="5">
    <tr>
      <th colspan="2">ĐĂNG NHẬP HỆ THỐNG</th>
    </tr>
    <tr>
      <td>Tên đăng nhập</td>
      <td><input name="TenDangNhap" type="text" size="25" /></td>
    </tr>
    <tr>
      <td>Mật khẩu</td>
      <td><input name="MatKhau" type="password" size="25" /></td>
    </tr>
    <tr>
      <td colspan="2" align="center">
        <input type="submit" value="Đăng nhập" />
      </td>
    </tr>
  </table>
</form>
```

5.1. Session

```
<?php
    session_start();
    $TenDN = $_POST['TenDangNhap'];
    $MKhau = $_POST['MatKhau'];
    if(trim($TenDN) == "")
        echo "Vui lòng nhập tên đăng nhập!";
    else if(trim($MKhau) == "")
        echo "Vui lòng nhập mật khẩu!";
    else if($TenDN == "demo" && $MKhau == "123456")
    {
        //thiết lập SESSION mã người dùng
        $_SESSION['MaND'] = "ND001";
        //thiết lập SESSION họ và tên
        $_SESSION['HoVaTen'] = "Trần Văn A";
        //Thiết lập SESSION quyền hạn, 1: admin, 2: user
        $_SESSION['QuyenHan'] = 1;
        Header("Location:quanly.php");
    }
    else
    {
        echo "Tên đăng nhập/mật khẩu không đúng <br />";
        echo "<a href = 'dangnhap.php'>Đăng nhập </a> <br />";
    }
?>
```

Code trang *dangnhap_submit.php*

5.1. Session

```
<?php
//dung $_SESSION để lấy $_SESSION đã thiết lập
if(isset($_SESSION['QuyenHan']) && $_SESSION['QuyenHan'] == 1)//Admin
{
    echo "Xin chào: ".$_SESSION['HoVaTen']."<br />";
    echo "<a href = 'dangxuat.php'>Đăng xuất </a> <br />";
    echo "<a href = '#'> Quản lý người dùng </a> <br />";
    echo "<a href = '#'> Quản lý bài viết </a> <br />";
    echo "<a href = '#'> Quản lý thay đổi hồ sơ </a> <br />";
    echo "<a href = '#'> Sao lưu & Phục hồi CSDL </a> <br />";
}
else if(isset($_SESSION['QuyenHan']) && $_SESSION['QuyenHan'] == 2)//User
{
    echo "Xin chào: ".$_SESSION['HoVaTen']."<br />";
    echo "<a href = 'dangxuat.php'>Đăng xuất </a> <br />";
    echo "<a href = '#'> Đăng bài viết </a> <br />";
    echo "<a href = '#'> Thay đổi hồ sơ </a> <br />";
}
else //chưa đăng nhập
{
    echo "Xin chào khách <br />";
    echo "<a href = 'dangnhap.php'>Đăng nhập </a> <br />";
}
?>
```

Code trang *quanly.php*

5.1. Session

```
<?php
    session_start();

    //HỦY bỏ SESSION
    unset($_SESSION['HoVaTen']);
    unset($_SESSION['QuyenHan']);

    Header("Location: quanly.php");
?>
```

Code trang *dangxuat.php*

5.2. Cookie

- ❖ Là một file nhỏ lưu các tùy chọn riêng của từng user được Server chỉ định lưu trữ trên Client và PHP có thể truy xuất tới được.
- ❖ Khi client có yêu cầu tới server thì browser sẽ gửi cookie ngược lên server, server sẽ đọc cookie này để kiểm tra, lấy thông tin và xử lý.
- ❖ Cookie được lưu trữ ở máy của client nên khi đóng trình duyệt thì cookie vẫn tồn tại và tồn tại bao lâu là do người dùng thiết lập. Ví dụ bạn thiết lập Cookie lưu trữ thông tin đăng nhập trong vòng 15 phút thì sau 15 phút nếu không có một thao tác thay đổi trên nó thì Cookie của bạn sẽ bị chết.
- ❖ **Cookie** thường được tạo ra khi người dùng truy cập một website, **cookie** sẽ ghi nhớ những thông tin như tên đăng nhập, mật khẩu, các tùy chọn do người dùng lựa chọn đi kèm. Các thông tin này được lưu trong máy tính để nhận biết người dùng khi truy cập vào một trang web.

5.2. Cookie

- ❖ Hai website khác nhau (cho dù cùng host trên 1 server) sẽ có 2 cookie khác nhau gửi tới browser.
- ❖ 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.
- ❖ Cookie ghi nhớ: đã login hay chưa, nhớ tên đăng nhập, mật khẩu, lần cuối cùng ghé thăm website,...
- ❖ Tính bảo mật thấp

5.2. Cookie

❖ So sánh Session và Cookie

| Cookie | Session |
|---|--|
| Cookie được lưu trữ trên trình duyệt của người dùng. | Session không được lưu trữ trên trình duyệt. |
| Dữ liệu cookie được lưu trữ ở phía client. | Dữ liệu session được lưu trữ ở phía server. |
| Dữ liệu cookie dễ dàng sửa đổi hoặc đánh cắp khi chúng được lưu trữ ở phía client. | Dữ liệu session không dễ dàng sửa đổi vì chúng được lưu trữ ở phía máy chủ. |
| Dữ liệu cookie có sẵn trong trình duyệt đến khi expired. | Sau khi đóng trình duyệt sẽ hết phiên làm việc (session) |

5.2. Cookie

❖ Khởi tạo cookie

setcookie("tên cookie", "giá trị", thời gian sống);

✚ Tên cookie: là tên đặt cho phiên làm việc.

✚ Giá trị: là thông số của tên cookie.

✚ Ví dụ:

setcookie("username", "admin", time() + 3600); // 1 giờ

❖ Lấy giá trị cookie:

\$_COOKIE["tên cookies"]

5.2. Cookie

❖ Hủy Cookie:

✚ Để hủy 1 cookie đã tạo có thể dùng 1 trong 2 cách:

- Gọi hàm setcookie có tham số là tên cookie

setcookie("Tên cookie")

- Đặt thời gian kết thúc cookie < 0.

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

```
<?php
    setcookie("name", "Kenny Huy", time()-3600);
?>
<html>
    <head>
        <title>Test page 1</title></head>
    <body>
        <b><a href="cookie2.php">Click here</a></b>
    </body>
</html>
```

5.2. Cookie

❖ Tạo trang *cookie.php* có nội dung sau:

```
<?php
    setcookie("name","Tran Van A",time() + 3600);
?>
<html>
    <head>
        <title>Test page 1</title></head>
    <body>
        <b><a href="cookie2.php">Click here</a></b>
    </body>
</html>
```

❖ Tạo trang *cookie2.php* có nội dung sau:

```
<html>
    <head><title>Result Page</title></head>
    <body>
        <?php
            if(isset($_COOKIE['name']))
                echo "Ten cua ban la <b>".$_COOKIE['name']. "</b>";
            else
                echo "Đã hủy cookie";
        ?>
    </body>
</html>
```