



CT428. LẬP TRÌNH WEB

CHƯƠNG 5. PHP & MYSQL (PHP)

Giảng viên: Trần Công Ân (tcan@cit.ctu.edu.vn)

Bộ môn Mạng máy tính & Truyền thông
Khoa Công Nghệ Thông Tin & Truyền Thông
Đại học Cần Thơ

2013 – 2014

PHP

NỘI DUNG

PHP LÀ GÌ?

CÚ PHÁP CƠ BẢN

HÀM (FUNCTION)

LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG (OOP)

PHP VÀ FORM

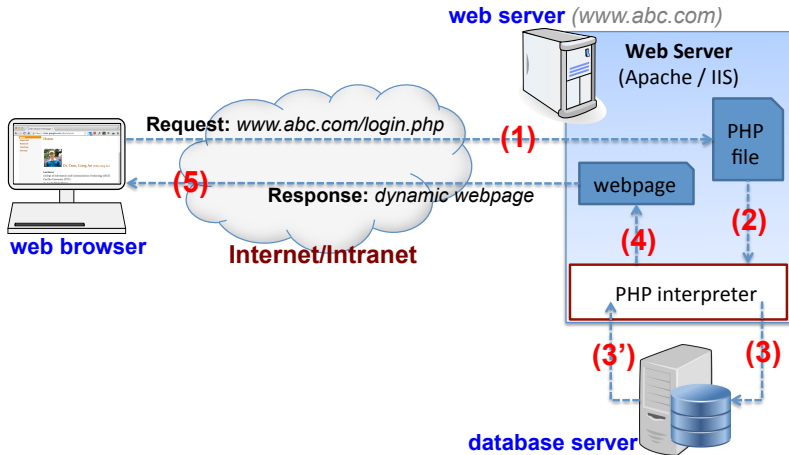
MySQL và PHP

COOKIE VÀ SESSION

PHP LÀ GÌ?

- ▶ PHP: **H**ypertext **P**re**P**rocessor (trình tiền xử lý siêu văn bản).
- ▶ Là một **ngôn ngữ script phía server**: được thực thi phía server và trả kết quả (**là 1 trang web HTML**) về cho browser.
- ▶ Sự thực thi PHP script không phụ thuộc vào web browser.
- ▶ Trình thông dịch PHP là phần mềm **mã nguồn mở, miễn phí**.
- ▶ Được hỗ trợ bởi hầu hết các web server (Apache, IIS, ...) và hệ điều hành thông dụng (Windows, Linux, MacOS, ...).
- ▶ Thường thực hiện các xử lý hướng nghiệp vụ.
- ▶ Một tập tin PHP có phần mở rộng là **.php**

CƠ CHẾ HOẠT ĐỘNG



CƠ CHẾ HOẠT ĐỘNG – Ví Dụ

~/htdocs/LTW428/vi-du-1.php

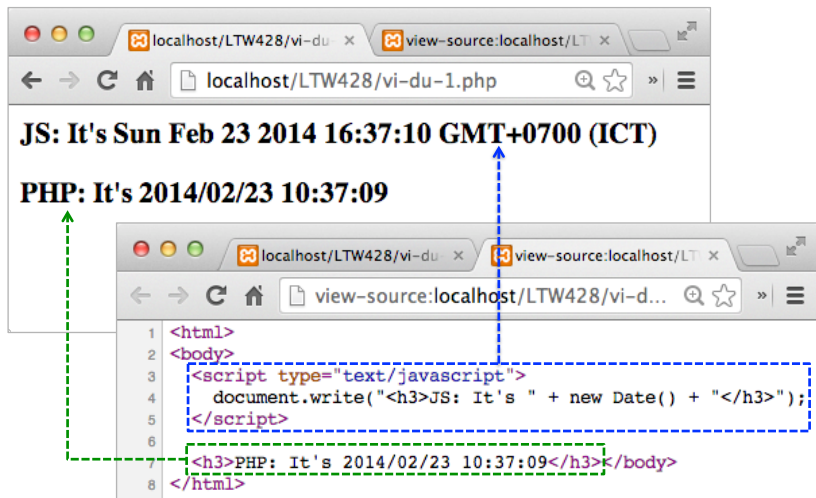
(http://localhost/LTW428/vi-du-1.php)

```
<html>
  <body>
    <script type="text/javascript">
      document.write("<h3>JS: It's " + new Date() + "</h3>");
    </script>

    <?php
      echo("<h3>PHP: It's " . date('Y/m/d H:i:s') . "</h3>");
    ?>
  </body>
</html>
```

✱ **Giải thích:** lệnh **echo** dùng để xuất 1 chuỗi về cho browser.

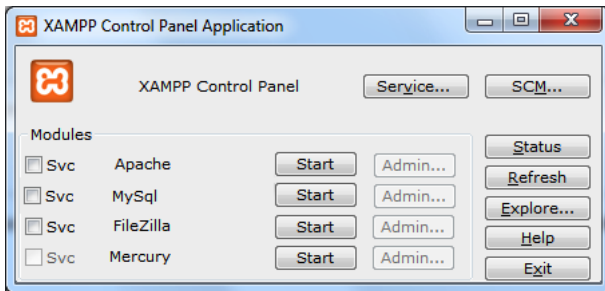
CƠ CHẾ HOẠT ĐỘNG – Ví Dụ



DOWNLOAD CÁC CHƯƠNG TRÌNH CẦN THIẾT

- ▶ **Apache** (Web server):
<http://httpd.apache.org/download.cgi>
- ▶ Chương trình **PHP**:
<http://www.php.net/downloads.php>
- ▶ **MySQL** DBMS:
<http://www.mysql.com/downloads/index.html>
- ▶ **XAMPP** – chương trình đóng gói cả 3 chương trình trên + Perl:
<http://www.apachefriends.org/download.html>
- ▶ Sử dụng XAMPP là phương pháp đơn giản nhất để cài đặt và thử nghiệm PHP & MySQL.

GIAO DIỆN CHƯƠNG TRÌNH XAMPP



- Giao diện chương trình XAMPP cho phép khởi động, dừng và cấu hình Apache và MySQL.

KIỂM TRA CÀI ĐẶT XAMPP

1. Khởi động chương trình Apache.
 2. Gõ vào thanh địa chỉ trình duyệt: **http://localhost**
- ✱ Thư mục mặc định chứa các web pages: **<XAMPP folder>[/xamppfiles]/htdocs/**



MỘT SỐ QUI ƯỚC

- ▶ Mã lệnh PHP được đặt một trong các tag:
 - ▶ `<? mã PHP ?>`
 - ▶ `<?php mã PHP ?>`
 - ▶ `<script language='php'> mã PHP </script>`
- ▶ Qui ước, cú pháp của PHP tương tự ngôn ngữ C và JavaScript.
 - ▶ Một câu lệnh kết thúc bằng dấu `;` và có thể nằm trên nhiều dòng.
 - ▶ Các lệnh không phân biệt chữ hoa, chữ thường.
 - ▶ Ghi chú cũng tương tự C và JS: gồm ghi chú 1 dòng (`//...`) và nhiều dòng (`/* ... */`)

BIẾN VÀ KIỂU DỮ LIỆU

- ▶ Kiểu dữ liệu: boolean (bool), integer (int), float, double, string.
- ▶ Kiểu dữ liệu phức: array, Object.
- ▶ Các kiểu đặc biệt: resource (3rd-party resources, e.g. DB), NULL.
- ▶ **Chú ý:** Các giá trị tương đương false: 0, 0.0, chuỗi rỗng, “0”, mảng rỗng, NULL.

BIẾN VÀ KIỂU DỮ LIỆU

► Biến:

- Bắt đầu bằng \$.
- Không cần khai báo biến – biến sẽ được tạo ở lần đầu gán giá trị.
- Kiểu biến sẽ được tự động gán, tùy vào dữ liệu của nó.
- Phân biệt chữ hoa, chữ thường.

► Phạm vi biến:

- Cục bộ: khai báo trong một hàm, chỉ t/xuất được bên trong hàm đó.
- Toàn cục: khai báo bên ngoài các khối lệnh, có thể t/cập từ bất kỳ vị trí nào trong chương trình (trong hàm phải dùng từ khóa **global**).

BIẾN VÀ KIỂU DỮ LIỆU

- ▶ **Biến tĩnh:** `static $var_name = value;`
 - ▶ Khai báo bên trong hàm (cục bộ)
 - ▶ Giá trị sẽ được lưu lại cho những lần gọi tiếp theo.
- ▶ Truy vấn kiểu dữ liệu của một biến: `gettype(var)`
- ▶ Kiểm tra biến và kiểu dữ liệu của biến: `is_bool()`, `is_int()`, `is_float()`, `is_double()`, `is_string()`, `is_object()`, `is_array()`, `is_numeric()`, `is_resource()`, `is_null()`, `isset()`, `empty()`.
- ▶ **Hằng số:** `define(const_name, value)`

BIẾN VÀ KIỂU DỮ LIỆU – Ví Dụ

```

<html>    <!-- datatype.php -->
  <body>
    <?php
      $f = 123.4;
      $s = "Hello world";
      $i = 100;
      echo("<p>");
      echo(gettype($f) . "<br>");    //double
      echo(var_dump($s) . "<br>");    //string(11) "Hello World"
      echo(is_int($i) . "<br>");      //1
      $i = 123.45;
      echo(var_dump($i));            //float(123.45)
      echo("</p>");
    ?>
  </body>
</html>

```



BIẾN TOÀN CỤC – VÍ DỤ

```
<?php    /* global-variable.php */

$a= 10; //global variable
function test() {
    echo "Inside function 1: " . $a; //NOTICE: Undefined variable a
    global $a;
    echo "Inside function 2: " . $a; //Inside function 2: 10
    echo "Inside function 3: " . $GLOBALS["a"];
}

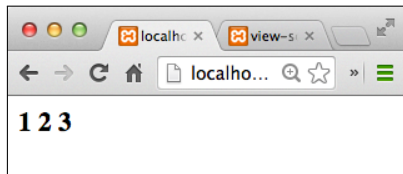
test(); //NOTICE (see above)
echo "Outside function: " . $a; //Outside function: 10

?>
```


BIẾN TÍNH – VÍ DỤ

```
<html>    <!-- static-variable.php -->
<body>
  <?php
    function test() {
      static $count = 1;
      echo $count++ . " ";
    }
    echo("<h3>");
    test(); test(); test();
    echo("</h3>");
  ?>

<body>
</html>
```



TOÁN TỬ

- ▶ Số học: +, -, /, %, ++, --
- ▶ Gán: =, +=, -=, *=, /=, %=
- ▶ So sánh: ==, ===, !=, !==, >, <, >=, <=
- ▶ Luận lý: &&, ||, !, xor
- ▶ Mảng (array): +, ==, , ===, !=, !==, <>
- ▶ Chuỗi: . (ghép chuỗi), .=

CHUỖI KÝ TỰ

- ▶ Một chuỗi ký tự có thể đặt trong dấu nháy đơn, kép hoặc **heredoc**.
- ▶ Dấu nháy đơn và heredoc: các ký tự escape sẽ không được thông dịch.

```
<?php /* string-specify.php */
    $name = "An";
    $s1 = "My name is: \"$name\"";
    $s2 = 'My name is: \"$name\"';
    $s3 = <<<_END
        My name is: \"$name"
    _END;
    echo $s1, "<br>";    //My name is: "An\"
    echo($s2 . "<br>"); //My name is: \"$name"
    echo($s3 . "<br>"); //My name is: \"An"
?>
```

MẢNG (ARRAY)

► Có 4 cách khai báo mảng:

► Mảng rỗng: `$arr_name = array();`

► Mảng + kích thước: `$arr_name = array(n);`

► Khai báo và khởi tạo: `$arr_name = array(val1, val2, ...)`

► Gán giá trị trực tiếp cho các phần tử của mảng:

`$arr_name[] = value;`

`$arr_name[index] = value;`

`$arr_name[] = value;`

`$arr_name[index] = value;`

`...`

`...`

ĐÁNH CHỈ SỐ CHO CÁC PHẦN TỬ MẢNG

- ▶ Có 2 cách đánh chỉ số cho các phần tử:
 - ▶ Dùng số thứ tự:

```
<?php
    $cities = array("Ho Chi Minh", "Ha Noi", "Can Tho", "Da Nang");
    echo $cities[0];      //output:Ho Chi Minh
?>
```

- ▶ Dùng định danh (ID/key):

```
<?php
    $cities2 = array("HCM" => "Ho Chi Minh", "HN" => "Ha Noi",
                    "CT" => "Can Tho", "DN" => "Da Nang"
                    );
    echo $cities2["CT"]; //output:Can Tho
?>
```

DUYỆT MẢNG BẰNG LỆNH foreach

► Cú pháp:

```
foreach ($array as $value) {  
    statements;  
}
```

```
foreach ($array as $key => $value) {  
    statements;  
}
```

► Ví dụ:

```
$arr = array(0, 2, 4);  
foreach ($arr as $val) {  
    echo $val . " ";  
} //output: 0; 2; 4;
```

```
$arr = array('a'=>4, 'c'=>6, 'b'=>8);  
foreach($arr as $key => $val) {  
    echo $key . ":" . $val . " ";  
} //output: a:4; c:6; b:8;
```

► Ngoài ra, ta có thể duyệt các phần tử của mảng dùng vòng lặp **thông qua chỉ số** của các phần tử.

MỘT SỐ HÀM LIÊN QUAN ĐẾN MẢNG

- ▶ `count($arr)`: đếm số phần tử của mảng.
- ▶ `sort/rsort($arr)`: sắp xếp mảng tăng/giảm dần.
- ▶ `max/min($arr)`: trả về phần tử lớn/nhỏ nhất.
- ▶ `array_push($arr, $element)`: thêm p/tử vào cuối mảng.
- ▶ `array_pop($arr)`: xóa phần tử ở cuối mảng.
- ▶ `array_unshift($arr, $element)`: thêm phần tử vào đầu mảng.
- ▶ `array_shift($arr)`: xóa phần tử ở đầu mảng.

Sinh viên tự xem thêm: mảng nhiều chiều và con trỏ mảng

LỆNH `echo` VÀ HÀM `print`

- ▶ Cả hai đều được sử dụng để **xuất 1 chuỗi về cho browser**.

- ▶ **Lệnh `echo`**: `echo(str);` hoặc `echo str[, str ...];`

```
echo("Hello ".$name);    hoặc    echo "Hello ", $name;
```

- ▶ **Hàm `print`**: `print(str);` hoặc `print str;`

```
print("Hello ".$name);    hoặc    print "Hello ".$name;
```

- ▶ Vì **`echo`** là một lệnh nên sẽ thực hiện nhanh hơn hàm **`print`**.
- ▶ Hàm **`print`** trả về `true` còn lệnh **`echo`** không có kết quả trả về.

```
($grade > 5) ? print("pass") : print("fail");
```

```
echo ($grade > 5) ? "pass" : "fail";
```


CẤU TRÚC ĐIỀU KHIỂN

- ▶ Cấu trúc điều khiển của PHP gần như tương tự JS (xem Ch4):
 - ▶ Rẽ nhánh: `if`, `if ...else`, `if ...elseif`, `?`, `switch ...case`
 - ▶ Lặp: `while`, `do ...while`, `for`, `foreach`
 - ▶ Các chỉ thị `break` và `continue`

HÀM (FUNCTION)

► Cú pháp:

```
function func_name([$arg1 [, $arg2...]]) {  
    statements;  
    :  
    :  
    [return return_val];  
}
```

- Tên hàm không phân biệt chữ hoa, chữ thường.
- Hàm có thể có giá trị trả về (dùng lệnh **return**) hay không có.

VÍ DỤ – HÀM GIẢI PHƯƠNG TRÌNH BẬC 1

```
<?php    /* function-ptb1.php */
function ptb1($a, $b) {
    if ($a == 0) {
        if ($b == 0)
            echo "Phuong trinh vo so nghiem";
        else
            echo "Phuong trinh vo nghiem";
    }
    else {
        echo "Nghiem x = " . $a/$b;
    }
}

ptb1(1, 2);    //output: Nghiem x = 0.5
ptb1(0, 2);    //output: Phuong trinh vo nghiem
ptb1(0, 0);    //output: Phuong trinh vo so nghiem
?>
```

HÀM – TRUYỀN THAM CHIẾU

- ▶ Truyền tham số bằng giá trị: hàm không thay đổi được giá trị của đối số.
- ▶ Truyền tham số bằng tham chiếu: hàm có thể thay đổi giá trị của đối số.

```
<?php
function add10(&$arg1, $arg2) {
    $arg1 += 10;
    $arg2 += 10;
}

$a = 10;
$b = 10;
add10($a, $b);
echo $a;    //output: 20
echo $b;    //output: 10
?>
```

```
<?php
function &func() {
    static $count = 0;
    $count++;
    return $count;
}

$var1 =& func();
echo $var1;    //output: 1
$var2 = func();
echo $var2;    //output: 2
echo $var1;    //output: 2

func();
func();
echo $var1;    //output: 4
echo $var2;    //output: 2
?>
```

HÀM – GÁN GIÁ TRỊ MẶC NHIÊN CHO ĐỐI SỐ

- Đối số của hàm có thể được gán giá trị mặc nhiên: Nếu lời gọi hàm không truyền giá trị cho đối số thì đối số sẽ nhận giá trị mặc nhiên.

```
<?php
function colorPar($par, $color="blue") {
    echo "<p style=\"color:$color;\">$par</p>";
}

colorPar("Hello");
//output: <p style="color:blue;">Hello</p>

colorPar("World", "red");
//output: <p style="color:red;">World</p>
?>
```

HÀM – THAO TÁC VỚI CÁC ĐỐI SỐ

- ▶ Một số hàm thao tác trên các đối số:
 - ▶ `int func_num_args(void)`: số các đ/số được truyền vào một hàm.
 - ▶ `mixed func_get_arg(int $arg_num)`: giá trị của một đối số trong danh sách các đối số.

```
<?php
function foo() {
    $argsnum = func_num_args();
    echo "Number of arguments: $argsnum; ";
    for ($i = 0; $i<$argsnum; $i++)
        echo func_get_arg($i) . ", ";
}

foo();    //output: Number of arguments: 0;
foo(1, 2); //output: Number of arguments: 2; 1, 2,
?>
```

HÀM XỬ LÝ CHUỖI

- ▶ `[l|r]trim($str [, $ch])`, `addslashes($str)`,
`stripslashes($str)`, `ucfirst($str)`, `ucwords($str)`,
`strtolower($str)`, `strtoupper($str)`, `strlen($str)`,
`strcmp($str1 , $str2)`, `strpos($str1 , $str2)`,
`str_replace($oldStr , $newStr, $str)` `strrev($str)`,
`explode($ch [, $str])`, `implode($str [, $ch])`

```
<?php
```

```
$str1 = "I'm a superman!";
```

```
$str2 = "<p style=\'color:blue;\'>";
```

```
echo addslashes($str1) . "\n"; //output: I\'m a superman!
```

```
echo $str2 . "\n"; //output: <p style=\'color:blue;\'>
```

```
echo stripslashes($str2) . "\n"; //output: <p style='color:blue;'
```

```
?>
```

HÀM TOÁN HỌC

- ▶ `abs($n)`, `sqrt($n)`, `pow($a, $b)`, `exp($x)`,
`range($start, $end)`, `floor($n)`, `ceil($n)`,
`is_nan($n)`, `number_format($n, $decimal=0,`
`$dec_point='.', $thou_sep=',')`

```
<?php
    $mang = range(1, 3);
    print_r($mang);    //output: Array ( [0] => 1, [1] => 2, [2] => 3 )

    $n = 1234.567;
    echo number_format($n, 2, ',', '.');    //output: 1.234,57
?>
```

- ▶ Tham khảo thêm:
http://www.w3schools.com/php/php_ref_math.asp

HÀM NGÀY GIỜ

- ▶ `getdate()`, `date($format_str)`, `checkdate($m, $d, $y)`,
`time()`, `date_add($date, $interval)`, `date_diff($date1,`
`$date2)`, `date_sub($date, $interval)`

```
<?php
echo date("d/m/Y");           //8/03/2014
echo date(DATE_RFC850);      //Saturday, 8-Mar-14 20:38:16 UTC

$d = getdate();               //return an array
echo $d["month"] . "-" . $d["year"]; //March-2014
?>
```

- ▶ Tham khảo thêm:
http://www.w3schools.com/php/php_ref_date.asp

OOP – KHAI BÁO LỚP

- ▶ Khai báo lớp:

```
class <classname> {  
    /* property + method declaration */  
};
```

- ▶ Khai báo thuộc tính (properties) tương tự như khai báo biến.
- ▶ Khai báo phương thức (methods) tương tự khai báo hàm.
- ▶ Thuộc tính truy cập: **public** | **protected** | **private**.
- ▶ Phương thức/thuộc tính tĩnh (static): **static**.
- ▶ Truy xuất p/thức bên trong p/thức: **\$this->method()**;
- ▶ Truy xuất t/tính bên trong p/thức: **\$this->property**;

OOP – Ví Dụ KHAI BÁO LỚP

```
<?php      /* Person.php */
class Person {
    public $id;
    public $name;
    public $dob;

    function displayInfo() {
        echo $this->id . " - " . $this->name . " - " . $this->dob;
    }

    function set($id, $name, $dob) {
        $this->id = $id;
        $this->name = $name;
        $this->dob = $dob;
    }
}

?>
```

OOP – TẠO ĐỐI TƯỢNG

- ▶ Cú pháp: `$<object_name> = new <class_name>;`
- ▶ Truy xuất t/tính của 1 đối tượng: `$object_name->property;`
- ▶ Truy xuất p/thức của 1 đối tượng: `$object_name->method();`

```
$tom = new Person();  
$tom->set("007", "Mr. Tom", date("d/m/Y"));  
$tom->displayInfo();    //output: 007 - Mr. Tom - 01/03/2014  
  
$tom->dob = "01/03/2010";  
$tom->displayInfo();    //output: 007 - Mr. Tom - 01/03/2010
```

OOP – CÁC THÀNH PHẦN TĨNH

- ▶ Là “**thành phần chung**” của tất cả các đối tượng của một lớp.
- ▶ Có thể **truy xuất thông qua lớp** (không cần tạo đối tượng).
- ▶ Truy xuất các thành phần tĩnh:
 - ▶ Bên trong lớp:
 - ▶ Thuộc tính: `self::$property`;
 - ▶ Phương thức: `self::method()...`;
 - ▶ Bên ngoài lớp:
 - ▶ Thuộc tính: `class_name::$property`;
 - ▶ Phương thức: `class_name::method()...`;

OOP – HÀM XÂY DỰNG

- ▶ Là một phương thức đặc biệt: tự động được gọi khi đối tượng được tạo ra.
- ▶ Dùng để gán giá trị khởi tạo cho các thuộc tính của đối tượng.
- ▶ Cú pháp:

```
public function __construct($arg1, $arg2,...) {  
  
    //initialisation  
  
}
```

OOP – HÀM XÂY DỰNG

```
<?php    /* Person-construct.php */
class Person {
    public $id;
    public $name;
    public $dob;

    protected static $count = 0;

    public function __construct($name) {
        $this->id = ++self::$count;
        $this->name = $name;
        $this->dob = @date("d/m/Y");
    }

    public function displayInfo() {...}
    public function set($id, $name, $dob) {...}
}
?>
```

```
$tom = new Person("Mr. Tom");
$tom->displayInfo();
//output: 1 - Mr. Tom - 07/03/2014

$jerry = new Person("Ms. Jerry");
$jerry->dob = "01/01/2002";
$jerry->displayInfo();
//output: 2 - Ms. Jerry - 01/01/2002

$test = @new Person(); //warning
$test->displayInfo();
//output: 3 - - 07/03/2014
```

THỪA KẾ

- ▶ Khai báo thừa kế: dùng từ khóa **extends**.
- ▶ Lớp con sẽ “thừa kế” tất cả các thành phần của lớp cha. Tuy nhiên, nó chỉ được truy xuất đến các thành phần **public** và **protected**.
- ▶ Truy cập đến các thành phần của lớp cha từ lớp con:
parent::property hoặc **parent::method()**.
- ▶ Lớp con có thể “khai báo chồng” (cùng tên) các thành phần của lớp cha.

THỪA KẾ

```
<?php    /* Student.php */

    require 'Person-construct.php';

    class Student extends Person {
        public $enroll;

        function __construct($name) {
            parent::__construct($name);
            $this->enroll = @date("d/m/Y");
        }

        public function displayInfo() {
            parent::displayInfo();
            echo " - " . $this->enroll;
        }
    }
?>
```

- **require**
'Person-construct.php': chèn đoạn mã trong tập tin Person.php vào.

```
$tom = new Student("Mr. Tom");
$tom->displayInfo();
//output:
// 1 - Mr. Tom - 07/03/2014 - 07/03/2014

$tom->set(7, $tom->name, "01/02/2002");
$tom->displayInfo();
//output:
// 7 - Mr. Tom - 01/02/2002 - 07/03/2014
```

MỘT SỐ VẤN ĐỀ KHÁC TRONG OOP

- ▶ Lớp ảo (abstract class).
- ▶ Ngăn chặn đè phương thức (final method).
- ▶ Giao diện (interface) và cài đặt giao diện (implementation).
- ▶ Xuất thông tin tự động cho đối tượng (`__toString()`).
- ▶ Các cách chèn đoạn mã trong các tập tin khác vào.



HTML FORM

- Form: là một t/phần của trang web, cho phép người dùng nhập liệu.

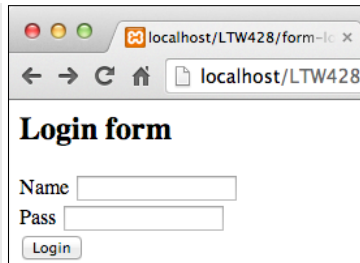
```
<form action="..." method="...">
  <!-- form controls -->
</form>
```

- action: **trang PHP** sẽ đáp ứng yêu cầu (xử lý d/liệu form + trả k/quả).
- method: POST | GET
 - POST: dữ liệu của form chèn bên trong yêu cầu HTTP.
 - GET: dữ liệu của form kèm theo URL (max 255B – 8K).

LẤY DỮ LIỆU GỬI VỀ TỪ FORM

- Dùng biến `$_GET` (p/thức GET), `$_POST` (p/thức POST) hoặc `$_REQUEST` (cả hai p/thức).

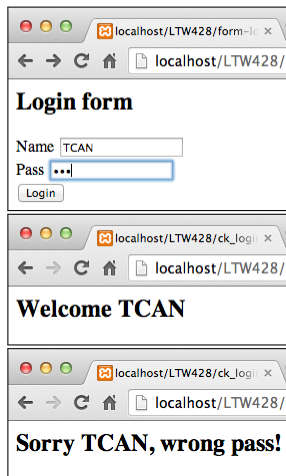
```
<html>           <!-- login.html -->
<body>
  <form action="ck_login.php" method="POST">
  <h2>Login form</h2>
  Name <input type="text" name="uname"/> <br>
  Pass <input type="password" name="pword"/>
  <br><input type="submit" value="Login"/>
  </form>
</body>
</html>
```



LẤY DỮ LIỆU GỬI VỀ TỪ FORM

```
<!-- ck_login.php -->

<html>
<body>
<h2>
<?php
    if ($_POST["pword"] == "abc")
        echo "Welcome ". $_POST["uname"];
    else
        echo "Sorry ". $_POST['uname']
            . ", wrong pass!";
?>
</h2>
</body>
</html>
```



KIỂM TRA SỰ TỒN TẠI CỦA DỮ LIỆU

- Kiểm tra sự tồn tại của một dữ liệu gửi từ form: hàm `isset()`.

```
<body>
<h3>
    <?php
        if (isset($_POST["uname"]) && isset($_POST["pword"])) {
            if ($_POST["pword"] == "abc")
                echo "Welcome ". $_POST["uname"];
            else
                echo "Sorry ". $_POST['uname'] . ", wrong pass!";
        }
        else {
            echo "<span style=\"color: red;\">"
                . "<i>uname</i> and <i>pword</i> are expected</span>";
        }
    ?>
</h3>
</body>
```

TRUY XUẤT DỮ LIỆU MySQL TRONG PHP

- ▶ Việc truy xuất dữ liệu MySQL từ PHP bao gồm các bước:
 1. Tạo nối kết đến MySQL.
 2. Chọn CSDL.
 3. Tạo câu truy vấn (DDL hoặc DML).
 4. Thực thi câu truy vấn.
 5. Nhận dữ liệu, xử lý dữ liệu và xuất ra trang web.
 6. Lặp lại các bước 4 và 5 cho đến khi truy cập hết các dữ liệu cần thiết.
 7. Đóng nối kết.

TẠO NỐI KẾT

► Tạo nối kết:

```
resource mysql_connect("host_name", "username", "password")
```

- **host_name**: địa chỉ của MySQL server.
 - **username**: tên người dùng CSDL.
 - **password**: mật khẩu người dùng CSDL.
- Thông thường, các thông số nối kết CSDL được lưu trong 1 tập tin riêng (.inc hoặc .php) và được chèn vào tập tin PHP cần truy xuất CSDL bằng lệnh **require_once 'filename'**.

TẠO NỐI KẾT

```
<?php    /* mysql-connect.php */
require_once 'connection.inc';

$db_server = @mysql_connect($hostname, $username, $password);
if (!$db_server) {
    echo "<h2>Unable connect to MySQL</h2>";
}
else {
    echo "<h2>Connect to MySQL!!!</h2>";
}
?>
```

```
<?php
/* connection.inc */
$hostname = "localhost";
$username = "root";
$password = "";
$database = "ltweb"

?>
```

KIỂM TRA VÀ XỬ LÝ LỖI

- ▶ Các cách kiểm tra và xử lý lỗi:
 - ▶ Dùng lệnh điều kiện (như trong VD trước).
 - ▶ Kết hợp lệnh `die(str)` và `mysql_error()`.



```
<?php    /* mysql-connect-die.php */
require_once 'connection.inc';
$db_server = @mysql_connect($hostname, $username, $password)
    or die("Unable connect to MySQL: " . mysql_error() . "<br/>");

echo "Connected to MySQL!!!<br/>";
?>
```

CHỌN CƠ SỞ DỮ LIỆU

► **Lệnh:** `int mysql_select_db("database_name", $connection)`

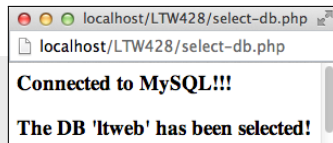
► `database_name`: tên CSDL cần chọn.

► `$connection`: 1 đối tượng connection.

```
<?php    /* select-db.php */
require "mysql-connect-die.php";

mysql_select_db($Database, $db_server) or
    die("Unable to select DB: " . mysql_error() . "<br/>");

echo "The DB '" . $database . "' has been selected!<br/>";
?>
```



TẠO VÀ THỰC THI CÂU TRUY VẤN

► Truy vấn CSDL: `resource mysql_query("query" [, $connection])`

- query: câu truy vấn.
- \$connection: đối tượng connection.

```
<?php    /* select-classics-table.php */
require "select-db.php";

$query = "SELECT * from classics";
$result = @mysql_query($query)
    or die("DB access error: " . mysql_error() . "<br/>");

echo "Query executed!"
?>
```



TRUY XUẤT DỮ LIỆU CỦA CÂU TRUY VẤN

- ▶ Các hàm sử dụng để truy xuất dữ liệu trả về trong câu truy vấn:
 - ▶ `mysql_num_rows($q_result)`: đếm số mẫu tin trong `$q_result`.
 - ▶ `mysql_fetch_fields($q_result)`: đếm số lượng trường.
 - ▶ `mysql_result($q_result, $row_index, "col_name")`: lấy giá trị 1 trường.
 - ▶ `mysql_fetch_row($q_result)`: trả về 1 mảng với giá trị các phần tử là giá trị của các trường trong bảng.
 - ▶ `mysql_fetch_array($q_result)` : tương tự hàm `mysql_fetch_row` với chỉ số mảng là tên trường.

TRUY XUẤT DỮ LIỆU CỦA CÂU TRUY VẤN – VD

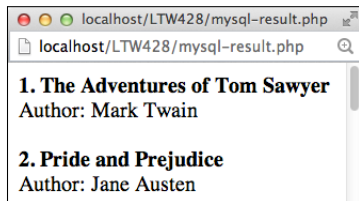
```
<?php    /* connect-select-db.php */  
    require_once 'connection.inc';  
  
    $conn = @mysql_connect($hostname, $username, $password)  
        or die("Unable connect to MySQL: " . mysql_error() . "<br>");  
  
    mysql_select_db($database, $conn) or  
        die("Unable to select DB: " . mysql_error() . "<br>");  
?>
```

mysql_result() - Ví Dụ

```
<?php    /* mysql-result.php */
    require_once 'connect-select-db.php';

    $query = "SELECT * from classics";
    $q_result = @mysql_query($query)
        or die("DB access error: " . mysql_error() . "<br>");

    $num_rows = mysql_num_rows($q_result);
    for ($i = 0; $i < $num_rows; $i++) {
        echo "<b>" . ($i+1) . ". ";
        echo mysql_result($q_result, $i, "title") . "</b><br>";
        echo "Author: " . mysql_result($q_result, $i, "author");
        echo "<br><br>";
    }
?>
```



mysql_fetch_row() – Ví Dụ

```
<?php    /* mysql-fetch-row */
    require_once 'connect-select-db.php';

    $query = "SELECT * from classics";
    $q_result = @mysql_query($query)
        or die("DB access error: " . mysql_error() . "<br>");

    $num_rows = mysql_num_rows($q_result);

    for ($i = 0; $i < $num_rows; $i++) {
        $row = mysql_fetch_row($q_result);
        echo "<b>" . ($i+1) . ". " . $row[1] . "</b><br>";
        echo "Author: " . $row[0] . "<br><br>";
    }
?>
```


mysql_fetch_array() - Ví Dụ

```
<?php      /* mysql-fetch-array.php */
require_once 'connect-select-db.php';

$query = "SELECT * from classics";
$q_result = @mysql_query($query)
    or die("DB access error: " . mysql_error() . "<br>");

$num_rows = mysql_num_rows($q_result);

for ($i = 0; $i < $num_rows; $i++) {
    $row = mysql_fetch_array($q_result);
    echo "<b>" . ($i+1) . ". " . $row["title"] . "</b><br>";
    echo "Author: " . $row[0] . "<br><br>";
}
?>
```

NGẮT NÓI KẾT – HÀM `mysql_close()`

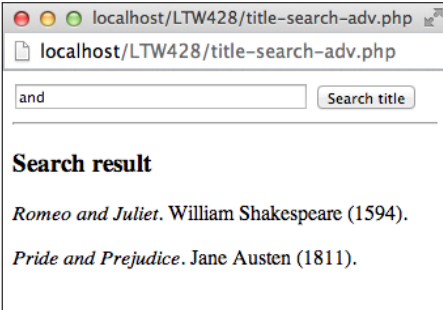
```
<?php    /* complete-skeleton.php */
//connect + select DB
$conn = @mysql_connect("hostname", "uname", "pword") or die("...");
@mysql_select_db("db_name", $conn) or die("...");

//create query string + execute query
$query = "SELECT * FROM table_name";
$q_result = @mysql_query($query) or die("...");

//use the data
while ($row = mysql_fetch_array($q_result)) {
    //...process the record...
}

//close the connection
mysql_close($conn);
?>
```

TÌM KIẾM SÁCH – MÔ TẢ



The screenshot shows a web browser window with the address bar displaying 'localhost/LTW428/title-search-adv.php'. The page content includes a search input field containing the text 'and' and a 'Search title' button. Below the search bar, the heading 'Search result' is followed by two lines of text: 'Romeo and Juliet. William Shakespeare (1594).' and 'Pride and Prejudice. Jane Austen (1811).'

localhost/LTW428/title-search-adv.php

localhost/LTW428/title-search-adv.php

and Search title

Search result

Romeo and Juliet. William Shakespeare (1594).

Pride and Prejudice. Jane Austen (1811).

TÌM KIẾM SÁCH – WEBPAGE & FORM

```
<html>          /* title-search.php */
  <body>
    <form action="title-search-adv.php" method="POST">
      <input type="text" size="40" name="search_kw"
        value="<? if (!empty($_POST['search_kw']))
          echo $_POST['search_kw'];?>" />
      <input type="submit" value="Search title">
    <hr></form>
    <h3>Search result</h3>
    <?php
      if (isset($_POST['search_kw'])) {
        include 'title-search-func-adv.php';
        search($_POST['search_kw']);
      }
    ?>
  </body>
</html>
```

TÌM KIẾM SÁCH – HÀM SEARCH

```
<?php    /* title-search-func.php */
function search($keyword) {
    require "connect-select-db.php";
    $keyword = trim($keyword);
    $new_kw = str_replace(" ", "% OR title LIKE '%", $keyword);
    $query = "SELECT * FROM classics WHERE title LIKE '%$new_kw%'";

    $result = mysql_query($query, $conn)
        or die ("DB accessed failed: " . mysql_error());

    while ($row = mysql_fetch_array($result))
        echo "<p><i>$row[title]</i>. $row[author] ($row[year]).";

    if (mysql_num_rows($result) == 0)
        echo "No title found";
}
?>
```

THÊM/XÓA SÁCH – MÔ TẢ

localhost/LTW428/title-manager.php

localhost/LTW428/title-manager.php

ISBN

Title

Author

Year

Category

Add Record

ISBN 9780099533474
 Title The Old Curiosity Shop
 Author Charles Dickens
 Year 1841
 Category Fiction
 DELETE

ISBN 9780192814968
 Title Romeo and Juliet
 Author William Shakespeare
 Year 1594
 Category Play
 DELETE

localhost/LTW428/title-manager.php

localhost/LTW428/title-manager.php

*Title '12345' has been added

ISBN

Title

Author

Year

Category

Add Record

ISBN 12345
 Title Lap trinh web
 Author ABC
 Year CS
 Category 2014
 DELETE

ISBN 9780099533474
 Title The Old Curiosity Shop
 Author Charles Dickens
 Year 1841
 Category Fiction
 DELETE

THÊM/XÓA SÁCH – MAIN PAGE

```
<?php    /* title-manager.php */

require "connect-select-db.php";    //connect to MySQL + select DB
require "title-delete-process.php"; //delete record if needed
require "title-add-process.php";    //add new record if needed
require "title-add-form.php";       //create [add title] form
require "title-delete-form.php";     //contains 'del_form_gen(...)'

//retrieve all records (to create [delete title] forms)
$query = "SELECT * FROM classics";
$result = mysql_query($query, $conn)
    or die ("DB Access error: " . mysql_error());

while ($row = mysql_fetch_array($result))
    del_form_gen($row); //generate [delete title] form

mysql_close($conn);    //close connection
?>
```

THÊM/XÓA SÁCH – KIỂM TRA XÓA SÁCH

```
<?php    /* title-delete-process */

    if (isset($_POST['delete']) && isset($_POST['isbn'])) {
        $isbn = $_POST['isbn'];
        $query = "DELETE FROM classics WHERE isbn='$isbn'";

        if (!mysql_query($query, $conn)) {
            echo "<h3> DELETE failed: " . $isbn . ". Error: "
                . mysql_error() . "</h3>";
        }
        else
            echo "*Title '$isbn' has been deleted<br>";
    }
?>
```


THÊM/XÓA SÁCH – KIỂM TRA THÊM SÁCH

```
<?php    /* title-add-process */

    if (isset($_POST["add"])) {
        $isbn = $_POST["isbn"];    $author = $_POST["author"];
        $title = $_POST["title"];    $year = $_POST["year"];
        $type = $_POST["type"];

        $query = "INSERT INTO classics VALUES " .
            "('$isbn', '$author', '$title', '$year', '$type', 0)";
        if (!mysql_query($query, $conn))
            echo "<h3>INSERT failed. " . mysql_error() . "</h3>";
        else
            echo "*Title '$isbn' has been added<br>";
    }
?>
```

THÊM/XÓA SÁCH – TẠO FORM THÊM SÁCH

```
<?php    /* title-add-form */
    echo <<<_ADD_TITLE_FORM
    <form action="title-manager.php" method="POST">
    <pre>
        ISBN <input type="text" name="isbn"/>
        Title <input type="text" name="title"/>
        Author <input type="text" name="author"/>
        Year <input type="text" name="year"/>
        Category <input type="text" name="type"/>
        <input type="submit" value="Add Record">
    </pre>
    <input type="hidden" name="add" value="yes">
    </form>
    _ADD_TITLE_FORM;
?>
```

THÊM/XÓA SÁCH – TẠO FORM XÓA SÁCH

```
<?php    /* title-delete-form */
function del_form_gen($row) {
    echo <<<_ADD_TITLE_FORM
    <form action="title-manager.php" method="POST">
    <pre>
        ISBN    $row[isbn]
        Title    $row[title]
        Author    $row[author]
        Year    $row[year]
        Category $row[type]
        <input type="submit" value="DELETE">
    </pre>
    <input type="hidden" name="delete" value="yes">
    <input type="hidden" name="isbn" value="$row[isbn]">
    </form>
    _ADD_TITLE_FORM;
} //add_form_gen()
?>
```

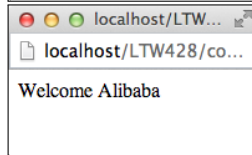
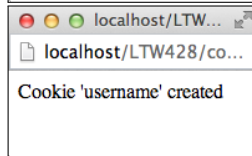
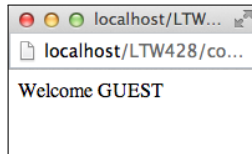
COOKIE

- ▶ Là một tập tin nhỏ được server/cầu lưu ở máy/tính người dùng web.
 - ▶ Tự động được gửi kèm theo các thông điệp HTTP request.
- ▶ Tạo cookie: `setcookie(name, value [, expire] ...)`
 - ▶ Hàm này phải được đặt trước thẻ `<html>`
- ▶ Truy xuất giá trị cookie: dùng biến `$_COOKIE`
- ▶ Hủy cookie:
 - ▶ Hết thời gian tồn tại (expired).
 - ▶ Nếu không có thời gian tồn tại, cookie sẽ bị hủy khi tắt browser.
 - ▶ Hủy tường minh: đặt expire của cookie là một giá trị trong quá khứ.

COOKIE – Ví Dụ

```
<?php
    setcookie("username", "Alibaba",
        time() + 60 * 10);
?>
<html> <body>
    <p>Cookie 'username' created</p>
</body> </html>
```

```
<html> <body>
    <?php
        if (isset($_COOKIE['username']))
            echo "Welcome $_COOKIE[username]";
        else
            echo "Welcome GUEST";
    ?>
</body> </html>
```



SESSION

- ▶ Là các biến chia sẻ giữa các trang web trong 1 ứng dụng web, tồn tại trên server
- ▶ Chỉ “nhìn thấy được” trong phiên làm việc của người dùng.
- ▶ Bắt đầu session: `session_start()`, đặt trước thẻ `<html>`.
- ▶ Truy xuất biến session: `$_SESSION['s_name']`
- ▶ Hủy session: `unset($session)` hoặc `session_destroy()`

SESSION

```
<?php    /* login.php */  
    //$login_ok = /* check the login information */;  
    if ($login_ok) {  
        session_start();  
        $_SESSION['logged_in'] = true;  
        /* redirect to other page */  
    }  
?>  
<html>  
    <!-- Login form! -->  
</html>
```

```
<?php    /* logout */  
    session_start();  
    if (isset($_SESSION['logged_in']))  
        unset($_SESSION['logged_in']);  
?>
```

SESSION

```
<?php      /* update-data.php */

    session_start();
    if (!isset($_SESSION['logged_in'])) {
        header('Location: login-session.php');
        die();
    }
?>

<html>
    <!-- update data form... -->
</html>
```

- ▶ Đoạn mã lệnh kiểm tra session được viết trong 1 tập tin riêng và được include vào các trang web cần k/tra đăng nhập.

