



THỰC HỌC – THỰC NGHIỆP



Conceive Design Implement Operate

LẬP TRÌNH PHP1

PHP & FORM

- ⊙ Dates
- ⊙ Variable testing
- ⊙ PHP và Form
- ⊙ GET & POST
- ⊙ REQUEST





PHẦN 1

- ❑ `date('Y-m-d H:i:s')`: ngày hiện hành.
- ❑ `date('Y-m-d H:i:s', time() - 60 * 60 * 24)`: ngày hôm qua
- ❑ `date('F j Y, H:i:s')`: các định dạng khác của ngày tháng năm (Tham khảo thêm <https://www.php.net/manual/en/function.date.php>)
- ❑ `time()`: thời gian hiện hành

- ❑ `date_parse`: thông tin cụ thể như năm, tháng, ngày, giờ, phút, giây phục vụ cho quá trình tính toán
 - ❖ Input: `$dateString`: Chuỗi thông tin ngày tháng cần lấy thông tin
 - ❖ Output: Trả về mảng chứa thông tin chi tiết về thời gian đã chọn

```
// Parse date: https://www.php.net/manual/en/function.date-parse.php  
$dateString = '2020-02-06 12:45:35';  
$parsedDate = date_parse($dateString);  
echo '<pre>';  
var_dump($parsedDate);  
echo '</pre>';
```

❑ `date_parse_from_format`: lấy thông tin về ngày cụ thể được truyền vào căn cứ vào kiểu định dạng của chuỗi thời gian.

❖ Input:

`$format`: Định dạng

`$dateString`: Chuỗi thời gian cần lấy thông tin

❖ Output: Trả về mảng chứa thông tin chi tiết về thời gian đã chọn

```
$dateString = 'February 4 2020 12:45:35';
```

```
$parsedDate = date_parse_from_format('F j Y H:i:s', $dateString);
```

```
echo '<pre>';
```

```
var_dump($parsedDate);
```

```
echo '</pre>';
```

- ❑ Biến toàn cục: có thể được truy cập từ bất cứ nơi nào trong tập lệnh PHP.
- ❑ Cách 1:

🐘 index.php

```
1  <?php
2  $a = 1; /* global scope */
3
4  function test()
5  {
6      |   echo $a; /* reference to local scope variable */
7  }
8
9  test();
10 ?>
11
```

□ Cách 2: sử dụng từ khoá `global`

 index.php

```
1  <?php
2  $a = 1;
3  $b = 2;
4
5  function Sum()
6  {
7      global $a, $b;
8
9      $b = $a + $b;
10 }
11
12 Sum();
13 echo $b;
14 ?>
15
```


- ❑ `$GLOBALS`: được sử dụng để truy xuất global variables từ bất kỳ đâu trong PHP script
- ❑ PHP lưu trữ tất cả các global variables trong mảng gọi là `$GLOBALS[index]` với index là tên của biến.
- ❑ Ví dụ

 index.php

```
1  <?php
2  $x = 75;
3  $y = 25;
4
5  function addition() {
6      |  $GLOBALS['z'] = $GLOBALS['x'] + $GLOBALS['y'];
7  }
8
9  addition();
10 echo $z;
11 ?>
```

- ❑ PHP xử lý dữ liệu do người dùng cung cấp. Dữ liệu cần được kiểm tra trước khi sử dụng để xác nhận rằng nó tồn tại và có giá trị hợp lệ. PHP cung cấp một số cấu trúc dựng sẵn được sử dụng cho mục đích này.
- ❑ `isset`: trả về `true` nếu biến tồn tại và đã được gán một giá trị khác `null`

```
isset($a); // false
```

```
$a = 10;
```

```
isset($a); // true
```

```
$a = null;
```

```
isset($a); // false
```

- ❑ `Empty`: kiểm tra xem biến được chỉ định có giá trị rỗng (empty) hay không – như là null, 0, false hoặc một chuỗi rỗng và trả về true nếu đúng.
- ❑ `Empty`: cũng trả về true nếu biến không tồn tại.

```
empty($b); // true
```

```
$b = false;
```

```
empty($b); // true
```

- ❑ `is_null` : được sử dụng để kiểm tra xem biến có được đặt là null hay không

```
// Prior to PHP 8
```

```
echo is_null($d); // true (undefined variable notice)
```

```
// As of PHP 8
```

```
echo is_null($d); // error (TypeError)
```

❑ unset: xoá biến khỏi phạm vi hiện tại.

```
$var = null; // free memory  
unset($var); // delete variable
```

❑ Null Coalescing Operator (??) là một dạng ngắn gọn của sử dụng toán tử bậc 3 với isset

```
$name = isset($x) ? $x : 'unknown';  
  
$x = null;  
$name = $x ?? 'unknown'; // "unknown"
```

- ❑ Null coalescing assignment operator (??=) : gán giá trị cho một biến chỉ khi biến đó chưa được gán (null). Nó cũng có thể được sử dụng với các biến không xác định.

```
// Assign value if $name is unassigned
```

```
$name ??= 'unknown';
```

```
// Same as above
```

```
if(!isset($name)) { $name = 'unknown'; }
```

Tên hàm	Mô tả
<code>Is_array()</code>	True nếu biến là 1 mảng
<code>Is_bool()</code>	True nếu biến là 1 bool
<code>Is_callable()</code>	True nếu biến có thể được gọi như 1 hàm
<code>Is_float(), is_double(), is_real()</code>	True nếu biến là 1 float
<code>Is_int(), is_integer(), is_long()</code>	True nếu biến là 1 integer
<code>Is_null()</code>	True nếu biến được thiết lập là null
<code>Is_numeric()</code>	True nếu biến là số hoặc chuỗi số
<code>Is_object()</code>	True nếu biến là 1 object
<code>Is_string()</code>	True nếu biến là 1 chuỗi

- ❑ **Variable information** : PHP xây dựng sẵn 3 hàm cho việc nhận thông tin về biến: `print_r`, `var_dump`, và `var_export`
- ❑ `print_r` hiển thị giá trị của biến theo cách mà con người có thể đọc được. Rất hữu ích cho mục đích debug.

```
$a = array('one', 'two', 'three');
print_r($a);
```

- ❑ Output với lệnh `print`

```
Array ( [0] => one [1] => two [2] => three )
```

- ❑ Output với lệnh `var_dump`

```
array(3) {
  [0]=> string(3) "one"
  [1]=> string(3) "two"
  [2]=> string(5) "three"
}
```


- ❑ HTML form có 2 thuộc tính bắt buộc: `action` và `method`
- ❑ Thuộc tính `action` chỉ định tập tin mà dữ liệu từ form được gửi đến khi form submit.
- ❑ Ví dụ phía dưới đây: form sẽ gửi input `myString` đến `mypage.php` khi form submit

```
<?php // myform.php ?>
<!doctype html>
<html>
<body>
    <form action="mypage.php" method="post">
        <input type="text" name="myString">
        <input type="submit">
    </form>
</body>
</html>
```

demo



PHẦN 2

- ❑ Thuộc tính bắt buộc thứ 2 của form là phương thức gửi, có 2 loại GET và POST
- ❑ Gửi dữ liệu với POST
 - ❖ Dữ liệu được gửi sẽ không hiển thị trên thanh url.
 - ❖ Khi form được submit, trình duyệt tải trang mypage.php và gửi dữ liệu của form.
 - ❖ Dữ liệu sẽ được nhận thông qua mảng `$_POST`. Tên của thuộc tính ở đầu vào của form thành khoá(key) trong mảng kết hợp.

```
<?php // mypage.php ?>
<!doctype html>
<html>
<body>
    <?php echo $_POST['myString']; ?>
</body>
</html>
```

❑ Gửi dữ liệu với GET

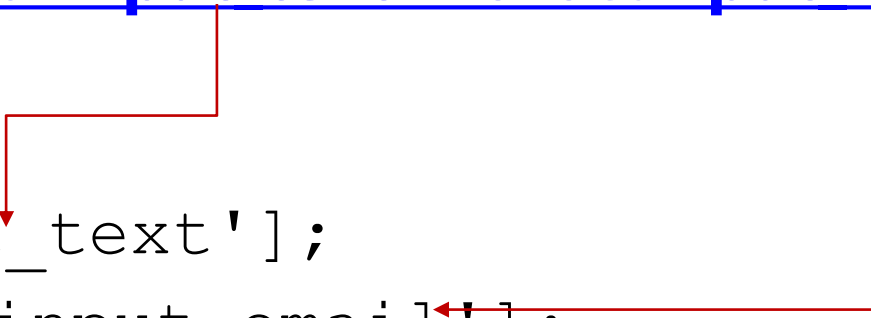
- ❖ Phương thức GET sẽ gửi dữ liệu trong url
- ❖ Dữ liệu gửi đi sẽ được nhận thông qua mảng \$_GET

```
// mypage.php  
echo $_GET['myString'];
```

- ❖ Cách đọc các GET variable từ URL

http://localhost/PHP1/display.php?input_text=hello&input_email=nona@gmail.com

```
$id_text    = $_GET['input_text'];  
$in_email   = $_GET['input_email'];
```



❑ Ví dụ

🐘 index.php ×

🐘 index.php

```
1  <form action="display.php" method="GET">
2      <label for="user">User
3      <input type="text" name="input_text">
4      </label>
5      <label for="user">Email
6      <input type="email" name="input_email">
7      </label>
8      <input type="submit" />
9  </form>
10
```

🐘 display.php ●

🐘 display.php

```
1  <?php
2
3  $id_text    = $_GET['input_text'];
4  $in_email   = $_GET['input_email'];
5
6  ?>
7
8
```

- ❑ Vì dữ liệu được chứa trong thanh địa chỉ, các biến không chỉ được chuyển qua form của HTML mà còn thông qua liên kết HTML => cách để chuyển biến từ trang này sang trang khác.
- ❑ Ví dụ

```
<?php // sender.php ?>
<!doctype html>
<html>
<body>
    <a href="receiver.php?myString=Foo+Bar">link</a>
</body>
</html>
```

❑ Ví dụ

```
<?php // receiver.php ?>
<!doctype html>
<html>
<body>
    <?php echo $_GET['myString']; // "Foo Bar" ?>
</body>
</html>
```


- ❑ Nếu việc gửi dữ liệu với `$_POST`, `$_GET` không quan trọng thì có thể sử dụng `$_REQUEST`.
- ❑ Mảng `$_REQUEST` chứa cả `$_GET` và `$_POST`. Ví dụ

```
<?php
if ($_REQUEST['currency']) # change currency on user request
{
    $currency = $_REQUEST['currency']; # use it
    setcookie('currency', $_REQUEST['currency'], 0, 'eshop.php'); # store it
}
else # use default currency
{
    $currency = 'USD';
}

# display shop contents with user selected currency
echo 'All prices are shown in ', $currency;

# let the user switch currency
echo '<a href="eshop.php?currency=USD">Switch to USD</a>';
echo '<a href="eshop.php?currency=EUR">Switch to EUR</a>';
?>
```

- ❑ Việc chỉnh sửa các thành phần của `$_POST` hoặc `$_GET` hoàn toàn không ảnh hưởng đến các thành phần của `$_REQUEST`

```
<?php
```

```
$_GET['foo'] = 'a';
```

```
$_POST['bar'] = 'b';
```

```
var_dump($_GET); // Element 'foo' is string(1) "a"
```

```
var_dump($_POST); // Element 'bar' is string(1) "b"
```

```
var_dump($_REQUEST); // Does not contain elements 'foo' or 'bar'
```

```
?>
```

❑ Mối quan tâm về bảo mật:

- ❖ Mọi dữ liệu do người dùng cung cấp đều có thể bị thao túng; Do đó cần được xác nhận và cải thiện (sanitized) trước khi sử dụng.
- ❖ **Validation (Xác thực)** có nghĩa là đảm bảo rằng dữ liệu cần đúng hình thức mình muốn như kiểm dữ liệu, phạm vi, nội dung.
- ❖ Ví dụ: xác thực địa chỉ email

```
if(!filter_var($_POST['email'], FILTER_VALIDATE_EMAIL))  
    echo "Invalid email address";
```

❑ Mối quan tâm về bảo mật:

- ❖ Dữ liệu do người dùng cung cấp dưới dạng text, `htmlspecialchars` sẽ được sử dụng.
- ❖ `htmlspecialchars` sẽ vô hiệu hoá bất kì đánh dấu HTML nào để đầu vào của người dùng được hiển thị nhưng không được diễn giải

```
// Sanitize for web page use  
echo htmlspecialchars($_POST['comment']);
```

❑ Submitting array

- ❖ Dữ liệu của form có thể nhóm thành mảng bằng cách sử dụng dấu ngoặc vuông [] phía sau tên biến trong form.
- ❖ Việc tạo mảng này có thể thực hiện đối với các input của form gồm `<input>`, `<select>` và `<textarea>`

```
<input type="text" name="myArr[]">
```

```
<input type="text" name="myArr[]">
```

- ❖ Có thể tạo key cho mảng các input

```
<input type="text" name="myArr[name]">
```

❑ Submitting array

- ❖ Lấy mảng dữ liệu được gửi từ form.

```
$val1 = $_POST['myArr'][0];  
$val2 = $_POST['myArr'][1];  
$name = $_POST['myArr']['name'];
```

- ❖ Ví dụ

```
<select name="myArr[]" size="3" multiple="true">  
  <option value="apple">Apple</option>  
  <option value="orange">Orange</option>  
  <option value="pear">Pear</option>  
</select>
```

```
foreach ($_POST['myArr'] as $item)  
  echo $item . ' '; // ex "apple orange pear"
```

demo

- ✓ Dates
- ✓ Variable testing
- ✓ PHP và Form
- ✓ GET & POST
- ✓ REQUEST



thank
you!