BÀI 4:
Định nghĩa function trong PHP
Xử lý lỗi và ngoại lệ trong PHP
Định dạng DateTime trong PHP

Mục tiêu bài học

- Định nghĩa function trong PHP
- Xử lý lỗi và ngoại lệ trong PHP
- Định dạng DateTime trong PHP

PHP functions

- Hàm (function) trong PHP tương tự như trong các ngôn ngữ lập trình khác. Hàm là một đoạn code được đóng gói thành một đơn vị độc lập.
- Lợi ích của hàm:
  - Hàm cho phép module hóa chương trình, làm cho việc phát triển chương trình trở nên dễ dàng hơn, đơn giản hơn.
  - Hàm giúp cho việc tìm lỗi hay bảo trì trở nên dễ hơn.
  - Hàm cho phép tái sử dụng: Một hàm chỉ cần định nghĩa một lần, sau đó có thể gọi và sử dụng nhiều lần.
- Có 2 loại hàm trong PHP
  - Các hàm được cung cấp sẵn bởi PHP (built-in functions).
  - Các hàm do người dùng tự định nghĩa

PHP functions

- Một số đặc điểm của hàm:
  - Hàm có thể nhận các tham số đầu vào, thực hiện xử lý nghiệp vụ và trả lại kết quả. Tham số khiến cho hàm trở nên linh động hơn. Các tham số của hàm là tùy chọn, có thể có hoặc không.
  - Để thoát ra khỏi hàm và trả lại giá trị từ hàm, ta sử dụng lệnh return.
  - Hàm không tự thực thi, hàm chỉ thực thi khi được gọi
  - Hàm cần phải được định nghĩa trước khi sử dụng.
- Để định nghĩa hàm trong PHP, ta sử dụng từ khóa function.

PHP functions

- Ví dụ:

```php
<?php
    function addFunction($num1, $num2) {
        $sum = $num1 + $num2;
        echo "Sum of the two numbers is : $sum";
    }

    addFunction(10, 20);
?>
```

- Kết quả:

```
Sum of the two numbers is : 30
```

PHP functions

- Trả lại giá trị từ hàm
  - Để trả lại giá trị từ hàm, ta sử dụng lệnh return.
  - Lệnh return sẽ dừng việc thực thi của hàm, và chuyển điều khiển về nơi gọi hàm.
- Ví dụ:

```php
<?php
   function addFunction($num1, $num2) {
      $sum = $num1 + $num2;
      return $sum;
   }
   $return_value = addFunction(10, 20);

   echo "Returned value from the function : $return_value";
?>
```

- Kết quả:

```
Returned value from the function : 30
```

PHP functions

- Thiết lập giá trị mặc định cho tham số
  - PHP cho phép ta thiết lập giá trị mặc định cho các tham số của hàm.
  - Tham số của hàm sẽ được gán giá trị mặc định, khi ta không truyền giá trị cho tham số khi gọi hàm.
- Ví dụ:

```php
<?php
function setHeight(int $minheight = 50) {
    echo "The height is : $minheight <br>";
}

setHeight(350);
setHeight();
setHeight(135);
setHeight(80);
?>
```

- Kết quả:

```
The height is : 350
The height is : 50
The height is : 135
The height is : 80
```

Demo

- Giáo viên quá trình định nghĩa và gọi hàm cho sinh viên.

Xử lý lỗi với PHP

- PHP cung cấp cơ chế xử lý lỗi mặc định, trong đó nếu có lỗi, chương trình sẽ cung cấp thông điệp lỗi bằng Tiếng Anh, trong đó có tên file phát sinh lỗi, mô tả ngắn gọn về lỗi, cùng với dòng lệnh xảy ra lỗi.
- Tuy nhiên, ta có thể tự định nghĩa một phương thức xử lý lỗi, để có thể cung cấp một thông điệp đơn giản và dễ hiểu cho người dùng để có thể nhận diện và xử lý lỗi dễ dàng hơn.
- Có 3 cách xử lý lỗi trong PHP:
  - Sử dụng hàm die()
  - Thiết lập trình xử lý lỗi (error handler) và bẫy lỗi (error triggering) do người dùng tự định nghĩa
  - Thông báo lỗi (error reporting)

Xử lý lỗi với PHP

- **Cách 1: Sử dụng hàm die()**
  Hàm die(): Dùng để xử lý và cung cấp những lỗi đơn giản trong PHP, VD khi chương trình không tìm thấy một file, một biến nào đó, ta có thể dùng hàm die() để in ra những thông báo lỗi đơn giản cho người dùng.
- Ví dụ không dùng hàm die():

```
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Demo error handling</title>
</head>
<body>
<?php

  $file=fopen("data.txt","r");

?>
</body>
</html>
```

- Kết quả:

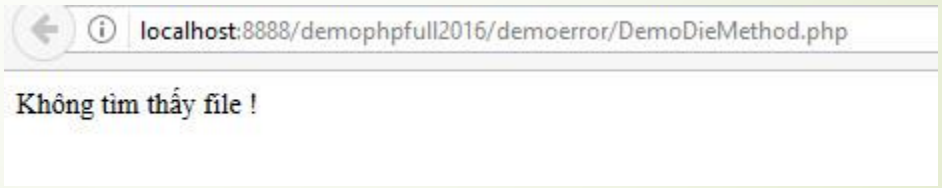
Warning: fopen(data.txt): failed to open stream: No such file or directory in **C:\xampp\htdocs\demophpfull2016\demoerror\DemoDieMethod.php** on line **10**

Xử lý lỗi với PHP

- Ví dụ có dùng hàm die():

```
<html>
<head>
<meta charset="utf-8">
<title>Demo error handling</title>
</head>
<body>
<?php
if(!file_exists("data.txt"))
  {
    die("Không tìm thấy file !");
  }
else
  {
  $file=fopen("data.txt","r");
  }
?>
</body>
</html>
```

- Kết quả:



Xử lý lỗi với PHP

- **Cách 2: Thiết lập trình xử lý lỗi (error handler) và trình bẫy lôi (error triggering)**
  - Phương pháp này được sử dụng để bẫy và chờ đón một lỗi khi người dùng không nhập đúng dữ liệu đầu vào.
  - Trong cách này, ta viết mã để định nghĩa một hàm xử lý lỗi (error handler), rồi thiết lập hàm xử lý lỗi bằng cách gọi hàm set_error_handler() của PHP.
  - Trong ví dụ sau, ta thấy biến $hoten chưa được khai báo mà đã sử dụng, nên sẽ có lỗi được ném ra.

Xử lý lỗi với PHP

- Ví dụ:

```php
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title>Demo Error Trigger</title>
</head>
<body>
<?php
//định nghĩa hàm xử lý lỗi (error handler function)
function ShowErrorHandler($errno, $errstr)
{
    echo "<b>Lỗi xảy ra:</b> [$errno] $errstr";
}


//thiết lập error handler
set_error_handler("ShowErrorHandler");

//dong lenh se phat sinh loi: bien $hoten chua duoc khai bao
echo($hoten);
?>
</body>
</html>
```

- Kết quả:

Xử lý lỗi với PHP

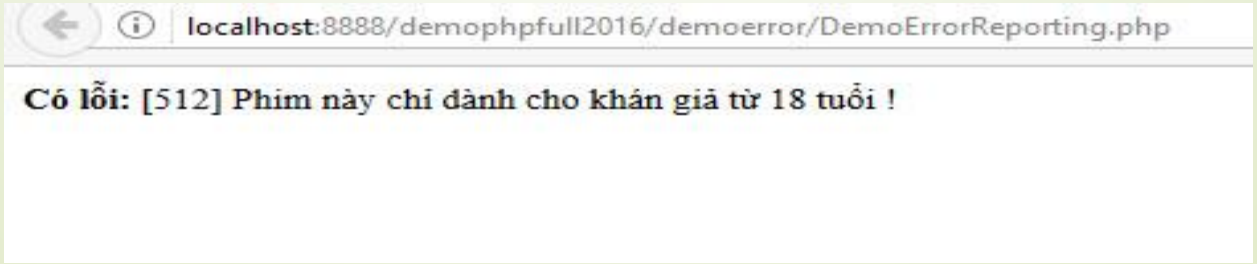
- **Cách 3: Thông báo lỗi (error reporting)**
  - Phương pháp này thông báo cho người dùng những thông điệp lỗi tùy biến dễ hiểu để người dùng dễ dàng phát hiện và xử lý lỗi, hoặc dữ liệu không như ta mong muốn.
  - Ví dụ:

```
<html>
<head>
<meta charset="utf-8">
<title>Demo Error Reporting</title>
</head>
<body>
<?php
//dinh nghia ham thong bao loi (error handler)
function thongbaoloi($errno, $errstr)
{
  echo "<b>Có lỗi:</b> [$errno] $errstr<br />";
  //ghi thong bao loi
  error_log("Thông báo lỗi: [$errno] $errstr");
}

//dang ky ham thong bao loi
set_error_handler("thongbaoloi", E_USER_WARNING);

//bay loi: trigger error
$tuoi= 12;
if ($tuoi < 18)
  {
  trigger_error("Phim này chỉ dành cho khán giả từ 18 tuổi !", E_USER_WARNING);
  }
?>
</body>
</html>
```

  - Kết quả:



Xử lý ngoại lệ với PHP

- Ngoại lệ (Exception): Là những lỗi xảy ra trong quá trình thực thi một chương trình, có thể làm thay đổi luồng thực thi của chương trình, hoặc khiến chương trình bị dừng một cách đột ngột không như mong muốn.
- Khi chương trình PHP có lỗi, ta gọi là chương trình đã ném (throw) ra ngoại lệ. Để tránh những ngoại lệ xảy ra, ta cần viết mã để bắt (catch) ngoại lệ.
- PHP cung cấp một cơ chế cho phép xử lý ngoại lệ, thông qua việc sử dụng các khối lệnh try-catch.

Xử lý ngoại lệ với PHP

- Khối try: Là khối lệnh dùng để thử lỗi. Trong khối try, ta sẽ đặt vào các đoạn mã có khả năng gây ra ngoại lệ, để kiểm tra xem có phát sinh ra ngoại lệ hay không.
- Khối try không thể sử dụng độc lập, mà sau nó cần phải có khối catch.
- Khối catch: Là khối lệnh dùng để xử lý (bắt) ngoại lệ. Trong khối catch, ta sẽ viết mã để xử lý ngoại lệ, đơn giản nhất là có thể in thông tin chi tiết về ngoại lệ đã xảy ra, để dễ dàng cho việc sửa lỗi.
- Mỗi khối catch sẽ có khai báo một biến, biến này dùng để chứa thông tin về ngoại lệ đã xảy ra, VD như: kiểu ngoại lệ, thông điệp mô tả ngoại lệ, dòng lệnh phát sinh ngoại lệ v.v…

Xử lý ngoại lệ với PHP

- **Cơ chế hoạt động của khối try-catch:**
  - Khi một dòng lệnh của khối try ném ra ngoại lệ, khối try sẽ dừng thực hiện tại dòng lệnh đó, tất cả các câu lệnh sau đó sẽ không được thực thi.
  - Chương trình sẽ chuyển đến khối catch để xử lý ngoại lệ. Nếu sau khối try có nhiều khối catch, chương trình sẽ lần lượt đánh giá từng khối catch, để xem khối catch nào tương thích với ngoại lệ, để thực thi nhằm xử lý ngoại lệ được ném ra.
  - Nếu khối try không ném ra ngoại lệ, chương trình sẽ bỏ qua, không thực hiện các khối catch.

Xử lý ngoại lệ với PHP

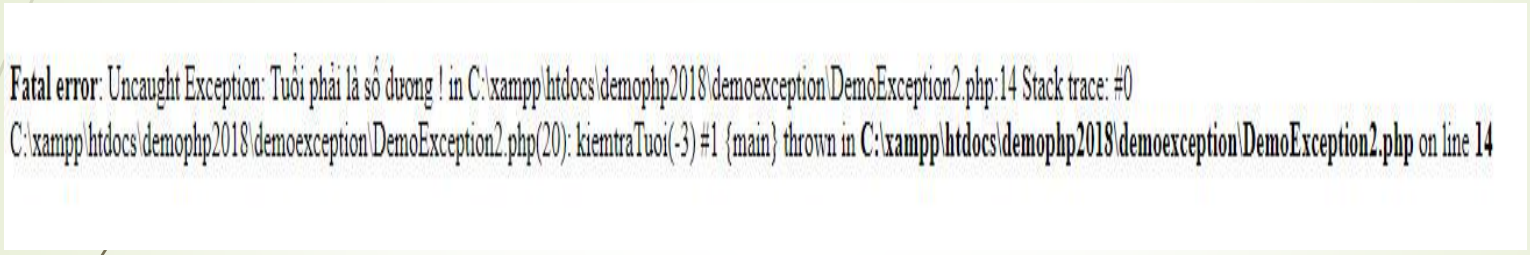
- Sau một khối try có thể có nhiều khối catch, trong trường hợp ta muốn xử lý nhiều loại ngoại lệ có thể xảy ra đối với mã nguồn.
  Để ném ra ngoại lệ một cách chủ động, ta sử dụng từ khóa throw.
- Nếu ta không xử lý ngoại lệ, khi có ngoại lệ xảy ra, chương trình sẽ ném ra một Fatal Error, cùng với thông báo Uncaught Exception như ví dụ sau:

```php
<?php
//create function with an exception
function kiemtraTuoi($age) {
  if($age < 0) {
    //nem ra ngoai le
    throw new Exception("Tuổi phải là số dương !");
  }
  return true;
}

//goi ham
kiemtraTuoi(-3);
?>
```

Xử lý ngoại lệ với PHP

- Ở đây, ta đã định nghĩa một hàm cho phép ném ra ngoại lệ khi tham số < 0. Sau đó ta gọi hàm để kiểm tra ngoại lệ. Chương trình này không xử lý ngoại lệ thông qua khối try-catch.
- Kết quả của chương trình như sau:



- Ta thấy thông báo lỗi rất dài và khó hiểu đối với người dùng như hình trên.

Xử lý ngoại lệ với PHP

- Để cung cấp thông báo lỗi thân thiện, cũng như để xử lý ngoại lệ, ta hãy khảo sát ví dụ dưới đây:

```php
<?php
//dinh nghia ham co nem ra ngoai le
function kiemtraTuoi($age) {
  if($age < 0) {
    //nem ra ngoai le
    throw new Exception("Tuổi phải là số dương !");
  }
  return true;
}
```

```php
//dinh nghia khoi try
try {
  kiemtraTuoi(-5);
  //in thong bao
  echo 'Kết thúc try';
}

//dinh nghia khoi catch
catch(Exception $e) {
  echo 'Ngoại lệ xảy ra: ' .$e->getMessage();
}
?>
```

Xử lý ngoại lệ với PHP

- Trong bài này, ta đã định nghĩa hàm giống bài ở trên, nhưng ta cung cấp cơ chế xử lý ngoại lệ, thông qua việc sử dụng khối try-catch. Trong hàm ta ném ra ngoại lệ, đồng thời truyền thông báo lỗi nếu tham số là số <0. Trong khối try, ta gọi hàm, đồng thời truyền giá trị < 0 vào hàm. Khối catch ta in thông báo lỗi xảy ra.
- Kết quả của chương trình như sau, thông báo lỗi được hiển thị cho người dùng như hình bên dưới:

Định dạng DateTime với PHP

- Hàm date() dùng để định dạng ngày tháng hoặc/và thời gian trong PHP.
- Cú pháp:

```
date(format,timestamp)
```

- Hàm date() có một tham số bắt buộc là định dạng ngày tháng (format)
- Ý nghĩa của các tham số:
  - format: Định dạng ngày tháng (required)
  - timestamp: Mặc định là thời gian hiện tại (optional)
- Ý nghĩa của một số định dạng:
  - d – Thể hiện ngày trong tháng (1-31)
  - m – Thể hiện tháng (1-12)
  - y- Thể hiện năm (4 chữ số)
  - l – Thể hiện ngày trong tuần
- Chú ý: Ta có thể chèn thêm các ký tự khác, VD như dấu (-), (/) hoặc (.) để cung cấp định dạng thân thiện hơn cho giá trị ngày tháng.

Định dạng DateTime với PHP

- Ví dụ:

```
<!DOCTYPE html>
<html>
<body>

<?php
echo "Today is " . date("Y/m/d") . "<br>";
echo "Today is " . date("Y.m.d") . "<br>";
echo "Today is " . date("Y-m-d") . "<br>";
echo "Today is " . date("l");
?>

</body>
</html>
```

- Kết quả:

Today is 2019/10/23
Today is 2019.10.23
Today is 2019-10-23
Today is Wednesday

Định dạng DateTime với PHP

- Hàm date() còn được sử dụng để lấy về thời gian hiện tại của hệ thống, cũng như định dạng thời gian.
- Một số ký tự dùng để định dạng thời gian:
  - H: Định dạng 24h (00-23)
  - h: Định dạng 12h (01-12)
  - i: Định dạng phút với một số 0 đứng trước (00-59)
  - s: Định dạng giây với một số 0 đứng trước (00-59)
  - a: Định dạng am hay pm

Định dạng DateTime với PHP

- Hàm date() còn được sử dụng để lấy về thời gian hiện tại của hệ thống, cũng như định dạng thời gian.
- Một số ký tự dùng để định dạng thời gian:
  - H: Định dạng 24h (00-23)
  - h: Định dạng 12h (01-12)
  - i: Định dạng phút với một số 0 đứng trước (00-59)
  - s: Định dạng giây với một số 0 đứng trước (00-59)
  - a: Định dạng am hay pm

Định dạng DateTime với PHP

- Ví dụ:

```
<!DOCTYPE html>
<html>
<body>

<?php
echo "Thời gian hiện tại là: " . date("h:i:sa");
?>

</body>
</html>
```

- Kết quả:

Thời gian hiện tại là: 01:59:09pm

Định dạng DateTime với PHP

- Để thiết lập time zone, ta sử dụng hàm sau:

```
<!DOCTYPE html>
<html>
<body>

<?php
date_default_timezone_set("Asian");
echo "The time is " . date("h:i:sa");
?>

</body>
</html>
```

- Kết quả:

The time is 02:00:42pm

Định dạng DateTime với PHP

- Hàm strtotime(): Cho phép chuyển đổi một chuỗi chứa giá trị ngày tháng trở thành một giá trị thời gian (ngày tháng).
- Ví dụ:

```
<!DOCTYPE html>
<html>
<body>

<?php
$d=strtotime("10:30pm April 15 2014");
echo "Created date is " . date("Y-m-d h:i:sa", $d);
?>

</body>
</html>
```

- Kết quả:

Created date is 2014-04-15 10:30:00pm

Định dạng DateTime với PHP

- Hàm strtotime() còn có thể cho phép chuyển đổi những chuỗi theo ngữ cảnh trở thành giá trị ngày tháng một cách rất thông minh và thuận tiện cho người dùng.
- Ví dụ:

```
<!DOCTYPE html>
<html>
<body>

<?php
$d=strtotime("tomorrow");
echo 'Ngày mai: ' . date("Y-m-d h:i:sa", $d) . "<br>";

$d=strtotime("next Sunday");
echo 'Chủ Nhật tới: ' . date("Y-m-d h:i:sa", $d) . "<br>";

$d=strtotime("+3 Months");
echo '3 tháng tới: ' . date("Y-m-d h:i:sa", $d) . "<br>";
?>

</body>
</html>
```

- Kết quả:

Tổng kết

- Định nghĩa function trong PHP
- Xử lý ngoại lệ trong PHP
- Định dạng DateTime trong PHP