Xử lý tập tin và thư mục

Nội dung

1. Tập tin
   - Mở tập tin
   - Đọc tập tin
   - Ghi tập tin
   - Đóng tập tin
   - Các hàm xử lý tập tin
2. Upload file
3. Thư mục

Mở tập tin

- Kết hợp kiểm tra file khi mở:

```
<?php
      //mở file nếu file hợp lệ, ngược lại thì thông báo lỗi
      $f = fopen(“hoa.txt”, “r”) or exit (“Không thể mở file!”);
?>
```

Mở tập tin

- Phân loại các chế độ mở file:
  - r: Chỉ đọc file. Bắt đầu đọc từ đầu file.
  - r+: Đọc và ghi. Bắt đầu từ đầu file.
  - w: Chỉ ghi file. Mở và xóa toàn bộ nội dung của file đã có hoặc tạo ra một file mới nếu file đó không tồn tại, sau đó ghi nội dung vào file.
  - w+: Đọc và ghi. Mở và xóa toàn bộ nội dung của file đã có hoặc tạo ra một file mới nếu file đó không tồn tại, sau đó ghi nội dung vào file.

Mở tập tin

- Phân loại các chế độ mở file
  - a: Ghi và đọc dữ liệu. Mở và ghi nội dung vào cuối file hoặc tạo ra một file mới nếu file không tồn tại.
  - a+: Ghi và đọc dữ liệu. Mở và ghi nội dung vào đầu file hoặc tạo ra một file mới nếu file không tồn tại.
  - x: Tạo và mở file để ghi. Tạo ra một file mới và ghi nội dung vào file. Nếu file đã tồn tại, trả về giá trị FALSE và thông báo lỗi.
  - x+: Tạo và mở để đọc và ghi file. Tạo ra một file mới. Nếu file đã tồn tại, trả về giá trị FALSE và thông báo lỗi.

Nội dung

1. Tập tin
   - Mở tập tin
   - Đọc tập tin
   - Ghi tập tin
   - Đóng tập tin
   - Các hàm xử lý tập tin
2. Upload file
   - Mở tập
   - Thư mục

Đọc tập tin

- Kiểm tra kết thúc tập tin
- Duyệt và đọc từng dòng nội dung trong tập tin
- Duyệt và đọc từng ký tự trong tập tin
- Đọc toàn bộ nội dung của tập tin

Đọc tập tin

- Kiểm tra kết thúc tập tin feof() (end of file)
  - Kiểm tra trạng thái cuối file
  - Sử dụng hàm này rất hữu hiệu cho việc đọc dữ liệu trong file khi không biết chiều dài cụ thể của file.
  - Kết quả trả về là TRUE nếu con trỏ ở cuối file.
  - Cú pháp: feof($f)
    - Tham số $f là biến khai báo ở hàm fopen().

Đọc tập tin

- Kiểm tra kết thúc tập tin feof() (end of file)
  - Chú ý: không thể đọc file khi mở file bằng chế độ w, a và x

```
<?php
    if (feof($f))
    {
        echo “Đã kết thúc file”;
    }
?>
```

Đọc tập tin

- Duyệt và đọc từng dòng nội dung trong tập tin
  - fgets() được dùng để đọc từng dòng nội dung trong file, có kết quả trả về là một chuỗi có chiều dài lên đến <độ_dài> -1 (độ dài có đơn vị tính là byte), nếu không quy định <độ_dài> thì hàm này sẽ tự thiết lập độ dài mặc định là 1024 byte.
  - Cú pháp: fgets($f [, int độ_dài])
    - Tham số $f là biến khai báo ở hàm fopen()

Đọc tập tin

- Duyệt và đọc từng dòng nội dung trong tập tin
  - Ví dụ: Đọc và in toàn bộ nội dung tập

```
<?php
    $f = fopen("hoa.txt", "r") or exit ("Không mở được file này ");
    while(!feof($f))
    {    $noi_dung = fgets($f);
         echo $noi_dung . "<br>";
    }
    fclose($f);
?>
```

Đọc tập tin

- Duyệt và đọc từng ký tự trong tập tin
  - fgetc(): đọc từng ký tự đơn trong file.
  - Chú ý: sau khi gọi hàm này thì con trỏ sẽ di chuyển tới ký tự tiếp theo trong file.
  - Cú pháp: fgetc($f)
    - Tham số $f là biến khai báo ở hàm fopen()

Đọc tập tin

- Duyệt và đọc từng ký tự trong tập tin

```
<?php
    $f = fopen(“hoa.txt”, “r”) or exit (“Không mở được file này ”);
    while(!feof($f))
    {    $noi_dung = fgetc($f);
         echo $noi_dung;
    }
    fclose($f);
?>
```

Đọc tập tin

- Đọc toàn bộ nội dung của tập tin
  - readfile(): đọc toàn bộ nội dung có trong file và hiển thị.
  - Kết quả trả về của hàm là số byte của file nếu đọc file thành công, ngược lại kết quả trả về là FALSE.
  - Cú pháp: readfile("đường dẫn và tên của file")

Nội dung

1. Tập tin
   - Mở tập tin
   - Đọc tập tin
   - Ghi tập tin
   - Đóng tập tin
   - Các hàm xử lý tập tin
2. Upload file
   - Mở tập
   - Thư mục

Ghi tập tin

- Ghi nội dung vào file
  - Cú pháp: fwrite(“đường dẫn và tên của file”, nội_dung [, độ_dài]);
  - **Chú ý:** độ dài là tham số tùy chọn: quy định số byte tối đa được ghi vào file của nội dung.

Ghi tập tin

```php
<?php
    $noi_dung = "Ghi nội dung vào tập tin";
    // thực hiện việc mở và ghi file
    $f = fopen("$ten_file", "a");
    fwrite($f,$noi_dung);
    fclose($f);
?>
```

Ghi tập tin

- Trước khi ghi chuỗi vào file cần phải định dạng lại chuỗi đó theo nhu cầu xuất dữ liệu trở lại khi đọc file.
- Cách thức định dạng: tự thiết lập. Tuy nhiên, có một số định dạng được quy định sẵn như sau:
  - \t: nhảy tab
  - \r\n: xuống dòng

Nội dung

1. Tập tin
   - Mở tập tin
   - Đọc tập tin
   - Ghi tập tin
   - Đóng tập tin
   - Các hàm xử lý tập tin
2. Upload file
   - Mở tập
   - Thư mục
3. Phần thứ 4

Đóng tập tin fclose()

- fclose() đóng tập tin đã mở
- Kết quả trả về là TRUE nếu đóng được file, ngược lại kết quả trả về là FALSE.
- Cú pháp: fclose($f);
  - Tham số $f là biến đã khai báo ở hàm fopen().

Nội dung

1. Tập tin
   - Mở tập tin
   - Đọc tập tin
   - Ghi tập tin
   - Đóng tập tin
   - Các hàm xử lý tập tin
2. Upload file
   - Mở tập
   - Thư mục

Các hàm xử lý tập tin

- Kiểm tra file tồn tại
- Lấy kích thước tập tin
- Xóa tập tin
- Kiểm tra tập tin

Kiểm tra file tồn tại

- Cú pháp:

  file_exists(“đường dẫn tập tin hoặc thư mục”)

- Kết quả trả về của hàm là TRUE nếu file tồn tại, ngược lại thì kết quả trả về là FALSE.

Kiểm tra tồn tại file_exists()

- Ví dụ: Kiểm tra sự tồn tại của file,

```
<?php
    $file = “du_lieu/khach_hang.txt”;
    if( !file_exists($file) )
    {   echo “Không có file này!”
        exit;
    }
    else
    {   echo readfile(file);
        echo “<br> Đọc file thành công”;
    }
?>
```

Lấy kích thước filesize()

- Cú pháp:

  filesize(“đường dẫn tập tin”)

  Kết quả trả về là kích thước của file (byte) nếu thành công, ngược lại thì trả về FALSE.

Lấy kích thước filesize()

- Ví dụ: Kích thước tập tin hoa.txt trong thư mục du_lieu

```
<?php
    echo filesize(“du_lieu/hoa.txt”);
    → 2104 byte
?>
```

Xóa tập tin unlink()

- Cú pháp:

  unlink("đường dẫn tập tin")

  Kết quả trả về:

  - TRUE nếu xóa file thành công
  - FALSE nếu không xóa được file

Xóa tập tin unlink()

```php
<?php
	$file = "du_lieu/san_pham.txt";
	if (!unlink($file))
		{
			echo "Không xóa được file: $file";
		}
	else
		{
			echo "Đã xóa file: $file";
		}
?>
```

Kiểm tra tập tin

- Cú pháp:
  is_file(“đường dẫn tập tin”)
  Kết quả trả về:
  - true: nếu đường dẫn tập tin tồn tại.
  - false: nếu đường dẫn tập tin không tồn
-

```
<?php
    echo is_file(“du_lieu/hoa.txt”);
    → 1: TRUE
?>
```

Nội dung

1. Tập tin
   - Mở tập tin
   - Đọc tập tin
   - Ghi tập tin
   - Đóng tập tin
   - Các hàm xử lý tập tin
2. Upload file

Giới thiệu

- Trong các ứng dụng Web, nhu cầu Upload File khá cần thiết => đối tượng FileFiled được sử dụng giúp người dùng có thể Upload File lên Web Server

Các bước Upload file

```
<form action="upload_file.php" method="post"
enctype="multipart/form-data">
<label for="file">Tên file:</label>
<input type="file" name="file_upload" id="file" />
<br />
<input type="submit" name="submit" value="Upload File" />
</form>
```

Các bước Upload file

- Bước 2: viết code để thực hiện việc

```
<?php
// kiểm tra file được Upload có hợp lệ hay không
…
// nếu hợp lệ: dùng hàm move_uploaded_file để di chuyển file upload vào thư mục định sẵn là thư mục upload
move_uploaded_file($_FILES["file_upload"]["tmp_name”], "upload/" . $_FILES["file_upload"]["name"]);
?>
```

Lấy thông tin file

- Sử dụng $_FILES
  - $_FILES["filefield"]["name"] – tên File
  - $_FILES[" filefield"]["type"] – kiểu của File
  - $_FILES[" filefield"]["size"] – kích cỡ của File
  - $_FILES[" filefield"]["tmp_name"] – tên tạm của File
  - $_FILES[" filefield"]["error"] – lỗi của File

Upload nhiều file

- Bước 1: Tạo Form Upload file

```
<form name="form1" method="post"
enctype="multipart/form-data">
    <input type="file" name="file[]" size="50"
multiple="true"  />
    <input type="submit" name="submit“
value="Upload File" />
</form>
```

Upload nhiều file

- Bước 2: viết code để thực hiện việc Upload File

```
//Danh sách dữ liệu file được chọn.
foreach ($_FILES['file']['name'] as $file) {
        $name[] = $file;
 }
 foreach ($_FILES['file']['tmp_name'] as $file){
        $tmp_name[] = $file;
 }
 foreach ($_FILES['file']['error'] as $file){
        $error[] = $file;
 }
```

Upload nhiều file

- Bước 2: viết code để thực hiện việc Upload File

```
  //kiểm tra tính hợp lệ của file
  //....
 //nếu hợp lệ dùng vòng lặp duyệt mảng file và dùng move_uploaded_file
để upload file lên server
$n=count($name);
for($i=0;$i<n;$i++)
{
      //nếu file hợp lệ thì làm.
       move_uploaded_file($tmp_name[$i], "upload/" . $name[$i])
}
```

Nội dung

1. Tập tin
   - Mở tập tin
   - Đọc tập tin
   - Ghi tập tin
   - Đóng tập tin
   - Các hàm xử lý tập tin
2. Upload file
3. Thư mục

3. Thư mục

- Mở thư mục
- Duyệt thư mục
- Đóng thư mục
- Tạo thư mục
- Kiểm tra đường dẫn thư mục

Mở thư mục

- Kết quả trả về sẽ là nguồn (chứa các thư mục và tập tin) của thư mục nếu thư mục mở thành công, ngược lại kết quả trả về sẽ là FALSE.
- Cú pháp: opendir("đường dẫn và tên thư mục
- Ví d

```
<?php
    $dir = opendir("du_lieu");
?>
```

Duyệt thư mục

- Duyệt thư mục bằng cách dùng vòng lặp while kết hợp với hàm readdir()
- Cú pháp: readdir("đường dẫn thư mục")
- Kết quả trả về của hàm là tên file hoặc thư mục trong thư mục được duyệt nếu đọc thành công. Ngược lại, kết quả trả về là FALSE

Duyệt thư mục

```
<?php   $dir = opendir("du_lieu");
      while (($file = readdir($dir)) !== false)
      {
            echo $file . "\t";
      }
      closedir($dir);
?>
.
..
chi_tiet_don_hang.txt
hoa.txt
khach_hang.txt
loai.txt
```

Đóng thư mục

- Cú pháp:

  closedir(“đường dẫn thư mục”);

- Ví dụ:

```
<?php
    closedir(“du_lieu”);
?>
```

Tạo thư mục

- Cú pháp: mkdir(“tên thư mục”);
- Kết quả trả về: là TRUE nếu tạo thư mục thành công, ngược lại kết quả trả về là FALSE.
- Ví dụ

```
<?php
    mkdir(“du_lieu”);
    → 1: TRUE
?>
```

Kiểm tra đường dẫn thư mục

- Cú pháp: is_dir("đường dẫn và tên thư mục");
- Kết quả trả về của hàm is_dir() là TRUE nếu thư mục tồn tại và là thư mục, ngược lại kết quả trả về là FALSE.
- Ví dụ: Kiểm tra sự tồn tại của thư mục du_lieu

```
<?php
    echo is_dir("du_lieu");
    → 1: TRUE
?>
```

Thảo luận