Bài 7

PHP nâng cao

Viện CNTT & TT

1. Session

- Khái niệm
- Cách thức hoạt động
- Khởi động session
- Đăng ký session
- Sử dụng session
- Hủy biến session

Khái niệm

- Session PHP cho phép lưu trữ thông tin người dùng trên trình duyệt (tên người dùng, danh mục hàng hóa, …) trong suốt quá trình làm việc của họ.
- Thông tin của session chỉ tạm thời và thông tin này sẽ bị xóa sau khi người dùng rời khỏi ứng dụng Web.
- Nếu cần, phải lưu trữ thông tin trong CSDL.

Cách thức hoạt động

- Session làm việc bằng cách tạo ra một địa chỉ duy nhất (UID) cho mỗi người sử dụng.
- UID có giá trị là một dãy số ngẫu nhiên.
- UID có thể được lưu trong COOKIE hoặc được truyền lên URL.
- Ngoài UID, có thể khai báo, khởi tạo và sử dụng một số biến session khác, tất cả các session này có giá trị cho mỗi người sử dụng khi họ truy cập đến ứng dụng Web.

Khởi động session

- Trước khi lưu trữ thông tin người dùng vào session, cần khởi động session.
- **Chú ý:** hàm khởi động session phải đặt phía trên thẻ HTML
- Cú pháp:

```
session_start();
```

Đăng ký session

- Sử dụng biến $_SESSION nhận và lưu trữ giá trị của biến session
- Cú pháp:

  $_SESSION[“tên biến session”] = “giá trị”;

- Ví dụ: tạo ra một biến session lưu tên đăng nhập của người dùng

```
<?php
        $_SESSION[“ten_dang_nhap”] = “phuong”;
?>
```

Sử dụng session

- Khi muốn sử dụng các biến session hoặc giá trị lưu trong biến session đã đăng ký => dùng biến $_SESSION.
- Cú pháp:

  $gia_tri = $_SESSION[“tên biến session”];

- Ví dụ: đọc giá trị biến session tên đăng nhập

```
<?php
        $ten_dang_nhap = $_SESSION[“ten_dang_nhap”];
        → phuong
?>
```

Hủy biến session

- Hủy toàn bộ các biến session:
  - Khi không cần dùng đến các biến session nữa thì có thể hủy bỏ toàn bộ các biến session đã đăng ký bằng hàm session_destroy()
- Cú pháp:

```
session_destroy();
```

Hủy biến session

- Hủy một biến session:
  - Khi không cần dùng đến biến session nào thì có thể dùng hàm unset() để hủy bỏ biến session đó.
- Cú pháp:

  unset($_SESSION[“tên biến session”]);
- Ví dụ: hủy bỏ biến session tên đăng nhập

```
<?php
        unset($_SESSION[“ten_dang_nhap”]);
?>
```

Ví dụ: Đếm số lần duyệt trang web

```
<?php
session_start( );
if (isset($_SESSION["count"]))
  $_SESSION["count"] = $_SESSION["count"] + 1;
else
  $_SESSION["count"] = 1;


print "Bạn đã truy cập trang này " . $_SESSION['count']
  . "lần.";
?>
```

Ví dụ: Ứng dụng cho Login

- Làm thế nào để ngăn không cho người dùng truy cập vào các trang web nếu chưa đăng nhập?
- Khi đăng nhập thành công thì chuyển sang trang khác không yêu cầu đăng nhập lại?
- Ý tưởng
  - Dùng các biến Session để lưu trạng thái đăng nhập của người dùng:

    `$_SESSION["IsLogin"]=true/false`: Lưu trạng thái đăng nhập

    `$_SESSION["Username"]`: Lưu Tên đăng nhập

    `$_SESSION["Authentication"]`: Lưu Loại quyền đăng nhập

Ví dụ: Ứng dụng cho Login

1. Tạo trang **login.htm** yêu cầu người dùng đăng nhập.
2. Tạo trang **validateuser.php** xử lí thông tin đăng nhập từ trang login.htm
   - Kết nối với **CSDL**, kiểm tra thông tin đăng nhập có hợp lệ hay không ?
     - Nếu không hợp lệ thì cho redirect về trang login.htm.
     - Nếu hợp lệ thì dùng một biến trong Session để lưu trạng thái login thành công
       - Ví dụ: `$_SESSION["IsLogin"] = true`.
       - **Lưu ý**: Phải đặt giá trị mặc định cho biến Session này là false khi khởi tạo một Session (xem ví dụ ở slide sau).
3. Tạo trang **logout.php** là trang xử lý khi người dùng logout
   - Reset trạng thái login là chưa đăng nhập (`$_SESSION["IsLogin"] = false`).

Ví dụ: Ứng dụng cho Login

4. Trong tất cả các trang muốn bảo mật, thêm đoạn mã sau để kiểm tra người dùng đã đăng nhập hay chưa, nếu chưa thì redirect lại trang login.htm.

```
<?php
session_start();

if ($_SESSION["IsLogin"] == false)
      header("Location: login.htm");
?>
```

2. Cookie

- Khái niệm
- Khai báo cookie
- Sử dụng cookie
- Hủy cookie

Khái niệm

- Được sử dụng để xác định thông tin của người dùng.
- Là một file nhỏ được server lưu trữ xuống từng máy tính của người dùng.
- Mỗi khi máy tính này yêu cầu một trang tới trình duyệt, nó cũng sẽ gửi theo cookie.
- Với PHP ta có thể tạo ra và sử dụng giá trị của biến cookie.

Cookie

- Trong Windows, Cookie được lưu ở thư mục Cookies.
- Chỉ chứa các thông tin đơn giản dạng name = value
- Sử dụng các giới hạn:
  - Expiration information (VD: 05/10/2005, 18:59:00 GMT),
  - Path information (VD: /user_section),
  - Domain information (VD: yourserver.com),
  - Secure parameter (HTTPS).
- Truy xuất thông qua biến toàn cục: **$_COOKIE[ ]**

Khai báo cookie

- **Chú ý**: Khai báo cookie ở phía trên thẻ HTML
- Cú pháp:

  setcookie(name, value, expire[, path, domain]);

  - name: tên biến cookie
  - value: giá trị
  - expire: thời gian giới hạn dành cho cookie – đơn vị tính là giây. Nếu thời gian này không được thiết lập trong hàm setcookie(), biến cookie này sẽ còn hiệu lực cho đến khi người dùng xóa tập tin cookie.
  - path: đường dẫn
  - domain: tên miền của website

Khai báo cookie

- Ví dụ: tạo ra một biến cookie người dùng có giá trị là "phuong", thời gian giới hạn là một giờ

```
<?php
        setcookie("nguoi_dung", "phuong", time()+3600);
?>
```

- Chú ý: giá trị của biến cookie sẽ tự động được URL mã hóa khi gửi cookie đi, và tự động giải mã khi nhận cookie về. (Nếu không muốn URL mã hóa thì dùng hàm setrawcookie())

Sử dụng cookie

- Dùng biến $_COOKIE để đọc giá trị biến cookie
- Cú pháp:

  $gia_tri = $_COOKIE[“tên biến cookie”];
- Ví dụ: đọc giá trị của biến cookie người dùng

```
<?php
        $nguoi_dung = $_COOKIE[“nguoi_dung”];
        →phuong
?>
```

Hủy cookie

- Khi muốn hủy một biến cookie thì cần kiểm tra lại thời gian giới hạn dành cho biến cookie này
- Sử dụng hàm setcookie() để hủy bằng cách đặt giá trị của biến cookie bằng “” và thời gian = - thời gian giới hạn
- Cú pháp:

  setcookie(name, “”, time() – thời gian giới hạn)

Hủy cookie

- Ví dụ: hủy biến cookie người dùng với thời gian giới hạn của biến này là 3600s

```
<?php
        $nguoi_dung = $_COOKIE[“nguoi_dung”];
        →phuong
?>
```

Gởi mail trong PHP

- Cú pháp:
  - **mail**(to, subject, message[, headers, parameters])
- Ý nghĩa các tham số:
  - to, subject, message: Như ý nghĩa các textbox khi soạn mail.
  - headers: Tùy chọn, có thể sử dụng Bcc, Cc.
  - parameters: Tùy chọn, các thông số về trình soạn, gởi mail.
  - Trong phần message: Sử dụng ký hiệu \n để xuống dòng.
- Lưu ý:
  - Bạn **không** thể mail từ localhost.
  - Muốn sử dụng chức năng gửi mail thì bạn phải có một host thực sự.

Gởi mail trong PHP

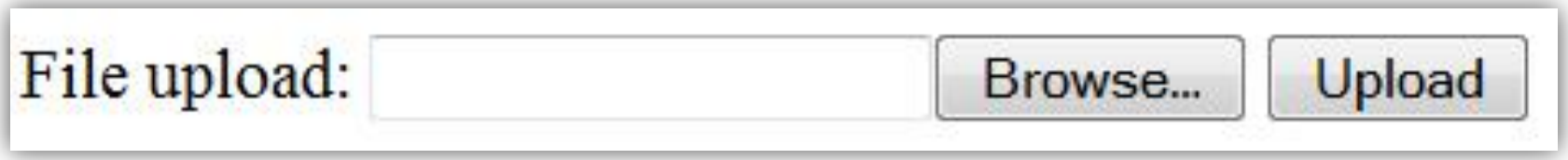
```
<?php
    $to         = 'hoangtung@inbox.com';
    $subject    = 'Chủ đề thư';
    $message    = 'Nội dung thông điệp';
    $headers    = 'From: webmaster@example.com'."\r\n".
        'Reply-To: webmaster@example.com'."\r\n".
        'X-Mailer: PHP/'.phpversion();

    mail($to, $subject, $message, $headers);
?>
```

- Lưu ý:
  - Vì lý do bảo mật nên một số host sẽ cấm sử dụng hàm mail của PHP.
  - Thường phối hợp với FORM để soạn thảo một trình gởi mail.

Upload File lên server

- Form upload:

```
<form action="upload.php" method="post"
                                enctype="multipart/form-data">
    File upload:
    <input type="file" name="myfile" />
    <input type="submit" value="Upload" />
</form>
```



- Lưu ý:
  - Luôn sử dụng phương thức POST.
  - Luôn sử dụng enctype="multipart/form-data" trong thẻ FORM.

Upload file lên Server

- Biến $_FILES[]:
  - $_FILES['myfile']['name']
    - Tên file đã được upload.
  - $_FILES['myfile']['type']
    - Kiểu file đã được upload, vd: image/gif, image/jpeg,...
  - $_FILES['myfile']['size']
    - Kích thước tập tin đã được upload (tính bằng bytes).
  - $_FILES['myfile']['tmp_name']
    - Vị trí file được lưu trữ tạm trên server.
  - $_FILES['myfile']['error']
    - Mã lỗi của việc upload (0 = Upload thành công).

Upload file lên Server

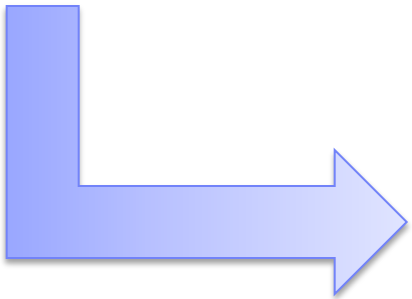
- Xử lý:

```php
<?php
    $dir = "data/"; //Upload vào thư mục data

    if($_FILES['myfile']['name'] != "")
    {
        $fileupload = $dir . $_FILES['myfile']['name'];
        if(move_uploaded_file($_FILES['myfile']['tmp_name'],
                                                $fileupload))
        {
            echo "Upload file thành công!";
        }
        else{
            echo "Upload file không thành công!";
        }
    }
    else{
        echo "Vui lòng chọn file để upload!";
    }
?>
```

Upload file lên Server

- Upload nhiều file:

```
<form action="upload.php" method="post"
                                enctype="multipart/form-data">
    Files upload:<br />
    File 1: <input type="file" name="myfile[]" /><br />
    File 2: <input type="file" name="myfile[]" /><br />
    File 3: <input type="file" name="myfile[]" /><br />
    <input type="submit" value="Upload" />
</form>
```



Upload file lên Server

- Xử lý upload nhiều file:

```
<?php
    foreach($_FILES['myfile']['error'] as $key => $error)
    {
        if($error == 0)
        {
            $tmp_name = $_FILES['myfile']['tmp_name'][$key];
            $name = $_FILES['myfile']['name'][$key];
            move_uploaded_file($tmp_name, "data/$name");
        }
    }
?>
```

Các hàm bảo mật trong chuỗi

- string addslashes(string $str)
- string stripslashes(string $str)
- string htmlspecialchars(string $str [, int $quote_style [, string $charset]])
- string md5(string $str [, bool $raw_output])
- string sha1(string $str [, bool $raw_output])

Các hàm bảo mật trong chuỗi

string addslashes(string $str)

- **Thêm** ký tự backslash (\) phía trước các ký tự ' " \ **NUL** trong chuỗi $str → Thường dùng trong các câu lệnh SQL để tránh xảy ra lỗi khi lưu và lấy dữ liệu từ CSDL.
- Các hàm tương tự:
  - string quotemeta(string $str)

```php
<?php
    $str = "I'dont know \'every thing\'";
    echo addslashes($str);
    //Output: I\'dont know \\\'every thing\\\'
?>
```

Các hàm bảo mật trong chuỗi

string stripslashes(string $str)

- **Xóa bỏ** ký tự backslash (\) xuất hiện trong chuỗi $str (ngược lại với hàm addslashes()).
- Hàm tương tự:
  - string **stripcslashes**(string $str)
-

```php
<?php
    $str = "I\'dont know \\\'every thing\\\'";
    echo addslashes($str);
    //Output: I'dont know \'every thing\'
?>
```

Các hàm bảo mật trong chuỗi

string htmlspecialchars(string $str [, int $quote_style [, string $charset]])

- Chuyển đổi các ký tự đặc biệt **&** **'** **"** **<** **>** trong chuỗi $str thành các thực thể HTML (Convert special characters to HTML entities).
- Khi đó:
  - **&** → &
  - **"** → " // khi không có **ENT_NOQUOTES**.
  - **'** → ' // khi có **ENT_QUOTES**.
  - **<** → <
  - **>** → >
- Ngược lại: htmlspecialchars_decode

Các hàm bảo mật trong chuỗi

string md5(string $str [, bool $raw_output])

- **MD5**: Message Digest 5 là một *hàm băm mật mã* được sử dụng phổ biến với giá trị băm dài **128** bit.
- Thường dùng để mã hóa mật khẩu, kiểm tra tính toàn vẹn của tập tin,…

```php
<?php
    $str = 'Lớp DH8TH';
    echo md5($str);
    //Output: 39a03156031b6a3ecf5dc5279ab3a77c
?>
```

Các hàm bảo mật trong chuỗi

string sha1(string $str [, bool $raw_output])

- Thuật toán **SHA-1** (Secure Hash Algorithm 1) tạo ra chuỗi mã băm có chiều dài cố định **160** bit từ chuỗi bit dữ liệu đầu vào $str có chiều dài tùy ý.
- Được sử dụng phổ biến và có công dụng như MD5, ngoài ra SHA-1 còn được sử dụng rất nhiều trong thương mại điện tử, tạo chữ ký số,…
- VD:

```php
<?php
    $str = 'Lớp DH8TH';
    echo sha1($str);
    //Output: a6fc5d0530e75a5288e4ff27b284741945677158
?>
```