PHP (Part 2)

4. Xử lý Form

4.1. Giới thiệu về form trong PHP

Form là một thành phần không thể thiếu trong lập trình Web bởi việc trao đổi và lấy thông tin từ người dùng để truyền tải lên Server phải thông qua Form. Nhìn chung Form có thể chia làm hai loại, một loại dùng để lấy thông tin bình thường và một loại dùng để lấy thông tin upload như upload hình ảnh, upload file word, ... nói chung là upload file.

 Khi gửi dữ liệu lên Server thì chúng ta sẽ thông qua hai phương thức đó là phương thức GET và phương thức POST, để hiểu sâu hơn 2 phương thức này chúng ta cần nắm được 2 khái niệm: Request và Response.

a. Request

Request có thể hiểu nhanh là thông tin gửi từ client lên server. Khi bạn lên trình duyệt browser gõ một địa chỉ nào đó, ví dụ bạn gõ là hoctot.cfi-elerning.edu.vn thì ngay lập tức trình duyệt sẽ dựa vào tên domain để gửi yêu cầu truy cập đến địa chỉ IP mà domain này đang trỏ tới (bạn đừng quan tâm đến domain này do ai quản lý), lúc này phía server sẽ phân tích yêu cầu và sẽ gửi luồng xử lý tới vị trí vị trí lưu trữ của mã nguồn PHP (hoặc mã nguồn bất kì) và nhiệm vụ của các mã nguồn là tiếp nhận yêu cầu, phân tích request đó và trả kết quả lại cho client.

Chúng ta có hai phương thức (2 cách) để gửi request từ client lên server đó là sử dụng phương thức GET và phương thức POST.

b. Response

Reponse là dữ liệu mà server trả về cho client. Ví dụ khi bạn nhập vào địa chỉ hoctot.cfi-elerning.edu.vn thì kết quả trả về (response) chính là giao diện của website và các thông tin của header. Như vậy dữ liệu mà server trả về là những đoạn mã HTML kèm theo các thông tin của header

4.2. Phương thức Get

Phương thức GET rất dễ nhận thấy đó là trên URL sẽ kèm theo dữ liệu mà chúng ta muốn gửi

Client gửi lên

Phương thức GET là phương thức gửi dữ liệu thông qua đường dẫn URL nằm trên thanh địa chỉ của Browser. Server sẽ nhận đường dẫn đó và phân tích trả về kết quả cho bạn. Server sẽ phân tích tất cả những thông tin đằng sau dấu hỏi (?) chính là phần dữ liệu mà Client gửi lên.

Để truyền nhiều dữ liệu lên Server ta dùng dấu & để phân cách giữa các cặp giá trị. Giả sử muốn truyền id = 12và title = ‘method\_get’ thì URL sẽ có dạng coffeecar.net?id=12&title=method\_get. Lưu ý với các bạn là vị trí các cặp giá trị không quan trọng, nghĩa là cặp title có thể nằm trước cặp id cũng được.

Server nhận dữ liệu

Tất cả các dữ liệu mà Client gửi lên bằng phương thức **GET** đều được lưu trong một biến toàn cục mà PHP tự tạo ra đó là biến **$\_GET**, biến này là kiểu mảng kết hợp lưu trữ danh sách dữ liệu từ client gửi lên theo quy luật key => value

4.2.1. Phương thức GET là gì?

Phương thức GET là hình thức client gửi dữ liệu lên server bằng cách bổ sung các tham số đằng sau URL mà ta hay gọi là Query String. Ví dụ ta có URL là: coffeecar.net?mail=hungcfi@gmail.com&domain= coffeecar.net thì lúc này phía server sẽ nhận được hai giá trị là:

Mail: hungcfi@gmail.com

Domain: coffeecar.net

Chính vì việc hiển thị dữ liệu trên URL một cách lộ liễu như vậy nên đối với những dữ liệu có tính bảo mật thì ta không thể sử dụng phương thức GET được. Thông thường những hành động nào có làm thay đổi database như thao tác thêm, xóa, sửa thì ta không nên sử dụng phương thức GET, còn những hành động nào không làm thay đổi database như thao tác lấy danh sách, tìm kiếm thì ta có thể sử dụng.

*Ví dụ: Sử dụng phương thức GET cho form tìm kiếm*

Bạn tạo một file tên là get.php với nội dung như sau:

<!DOCTYPE html>

<html>

 <head>

 <title>Xử lý form với GET</title>

 <meta charset="UTF-8">

 <meta name="viewport" content="width=device-width, initial-scale=1.0">

 </head>

 <body>

 <form method="get" action="get.php">

 <input type="text" name="q" value=""/>

 <input type="submit" name="btn" value="Search"/>

 </form>

 </body>

</html>

*Chú ý:*

* Đối với form action="get.php" và method="get"
* Đối với ô nhập dữ liệu có name="q"
* Đối với button có name="btn" và value="Search"

Bạn chạy file này lên sau đó nhập dữ liệu tìm kiếm vào rồi nhấn button Search thì lúc này giao diện và URL sẽ như sau:

<http://localhost/get.php?q=H%C3%B9ng&btn=Search>

Như vậy phương thức GET trong form thực chất là sự chuyển hướng trang kèm theo việc bổ sung phần query string cho URL. Như ví dụ trên khi ta tạo form với phương thức GET thì dữ liệu sẽ đưa lên URL với key chính là tên của ô input và value chính là dữ liệu của ô input.

4.2.2. Lấy dữ liệu từ phương thức GET

Để lấy dữ liệu từ phương thức GET thì ta phải sử dụng biến $\_GET, đây là biến toàn cục lưu trữ các dữ liệu từ client gửi lên server thông qua phương thức GET.

Quay lại ví dụ ở trên thì bạn thay đổi mã code cho file get.php như sau:

<!DOCTYPE html>

<html>

 <head>

 <title>Xử lý form với GET</title>

 <meta charset="UTF-8">

 <meta name="viewport" content="width=device-width, initial-scale=1.0">

 </head>

 <body>

 <form method="get" action="get.php">

 Họ và tên: <input type="text" name="ten" value=""/><br><br>

 Giới tính : <input type="text" name='gt' value=''>

 <input type="submit" name="btn" value="Search"/>

 </form>

 <?php

 **var\_dump($\_GET);**

 ?>

 </body>

</html>

Bây giờ bạn hãy chạy lại file get.php thì dễ dàng thấy biến $\_GET lúc này là một mảng rổng.

Bạn hãy nhập vào tên: Trần Sơn và giới tính: Nam và click nút Search, sẽ được 1 mảng có 3 phần tử như hình (hàm **var\_dump($\_GET);** sẽ xuất dữ liệu của biến $\_GET ra màn hình)



Ví dụ dùng phương thức GET

Viết một ứng dụng nhập vào năm sinh của bạn và tính ra số tuổi hiện tại. Chúng ta tạo file tuoi.php

Trước khi giải bài này thì cần phải phân tích thuật toán. Như đề bài ta nhập vào năm sinh thì muốn biết số tuổi ta chỉ cần lấy năm hiện tại trừ đi số năm sinh mà người dùng nhập vào là ra. Để lấy năm hiện tại thì ta dùng hàm xử lý ngày tháng trong PHP đó là hàm date().

Hàm date()

*Có cú pháp*

date ( $format, $timestamp = ' time() ')

*Trong đó:*

* $format là định dạng mà hàm này sẽ trả về
* $timestamp là thời gian truyền vào (kiểu INT), mặc định nó sẽ lấy thời gian hiện tại (chính là hàm time() ).

Danh sách các $format thường dùng:

* d: trả về ngày tháng (số)
* D: trả về ngày của tháng (tiếng Anh)
* m: trả về tháng của năm (số)
* M: Trả về tháng của năm (tiếng Anh)
* y: trả trả về năm (2 số cuối của năm)
* Y: trả về năm đầy đủ 4 số
* H: trả về số giờ (kiểu 24h)
* h: trả về số giờ (kiểu 6h)
* i: trả về số phút
* s: trả về số giây

Ví dụ: Lấy thời gian hiện tại theo định dạng ngày/tháng/năm - giờ:phút:giây

Chiếu theo các định dạng ở danh sách trên thì ta sẽ có chuỗi format như sau: d/m/Y - H:i:s

Ta có mã code PHP như sau:

echo date('d/m/Y - H:i:s');

Chúng ta có code của toàn bộ chương trình như sau:

<!DOCTYPE html>

<html>

 <head>

 <title>Xử lý form với GET</title>

 <meta charset="UTF-8">

 <meta name="viewport" content="width=device-width, initial-scale=1.0">

 </head>

 <body>

 <h1>ứng dụng tìm năm sinh</h1>

 <form method="get" action="tuoi.php">

 <input type="text" name="year" value=""/>

 <input type="submit" name="btn" value="Tìm Tuổi"/>

 </form>

 **<?php**

 if (!empty($\_GET['btn'])){

 // Bước 1: Lấy thông tin

 **$year = isset($\_GET['year']) ? (int)$\_GET['year'] : 0;** *// nếu có năm sinh nhập vào thì lấy năm (ép kiểu int) ngược lại cho bằng 0*

 // Bước 2: Lấy năm hiện tại

 $current\_year = date('Y');

 // Bước 2: Kiểm tra năm sinh hợp lệ hay không, nếu không thì thông báo

 if ($year <= 0){

 echo 'Năm sinh bạn nhập không hợp lệ';

 }

 else if ($year > $current\_year){

 echo 'Năm sinh bạn nhập lớn hơn năm hiện tại';

 }

 else{

 echo 'Số tuổi của bạn là ' . ($current\_year - $year);

 }

 }

 **?>**

 </body>

</html>

Trong code này chúng ta còn dùng thêm hàm isset() của PHP.

 Hàm isset()

Hàm isset() được dùng để kiểm tra một biến nào đó đã được khởi tạo trong bộ nhớ của máy tính hay chưa, nếu nó đã khởi tạo (tồn tại) thì sẽ trả về TRUE và ngược lại sẽ trả về FALSE.

*Ví dụ: Kiểm tra biến $domain có tồn tại hay không*

if (isset($domain)){

 echo 'Biến domain đã tồn tại';

}

else{

 echo 'Biến domain chưa tồn tại';

}

Nếu trong quá trình biên dịch nếu trong code có sử dụng một biến không tồn tại thì trình biên dịch sẽ ngưng xử lý và thông báo lỗi ngay, chính vì vậy thông thường những trường hợp mà ta không chắc chắn là biến đó luôn tồn tại thì trước khi sử dụng hãy kiểm tra nó.

Thông thường khi lấy thông tin từ FORM (xem bài post và get trong php) thì bạn nên kiểm tra nó có tồn tại không rồi hãy lấy, nếu không người dùng sẽ sử dụng firebug đổi một số name của các thẻ input thì chương trình sẽ lỗi ngay.

*Ví dụ*

if (isset($\_POST['submit']))

{

 $fullname = isset($\_POST['fullname']) ? $\_POST['fullname'] : '';

 $address = isset($\_POST['address']) ? $\_POST['address'] : '';

 $email = isset($\_POST['email']) ? $\_POST['email'] : '';

 $phone = isset($\_POST['phone']) ? $\_POST['phone'] : '';

}

*Trong code này chúng ta thấy xuất hiện toán tử ? vậy nó là toán tử gì. Khi làm việc với các framework PHP, chúng ta thấy xuất hiện các toán tử ?? và ?: được sử dụng rất thường xuyên.*

Hàm empty()

Hàm **empty()** trong php dùng để kiểm tra một biến nào đó có giá trị rỗng hoặc chưa được khởi tạo hay không.

Giả sử ta có biến $var và giá trị của nó sẽ là rỗng nếu nó nằm một trong các trường hợp sau:

* $var = 0 hoặc $var = '0'
* $var = NULL
* $var = '';
* $var = FALSE
* $var *không tồn tại*

Như vậy có 6 trường hợp hàm empty() sẽ đúng. Tuy nhiên trường hợp cuối cùng là $var không tồn tại là hơi đặc biệt một chút, vấn đề này rất giống với hàm isset().

Toán tử ternary (ba ngôi)

Toán tử điều kiện (ba ngôi) là toán tử duy nhất của PHP (hoặc JavaScript) cần tới ba toán hạng. Toán tử này thường dùng tắt cho câu lệnh if.

*Cú pháp*

**condition** **?** **exprT** **:** **exprF;**

*Trong đó*

* **Condition***:* Biểu thức mà giá trị của nó được dùng như điều kiện.
* **exprT** **:** **exprF :** Biểu thức với giá trị có kiểu bất kỳ.

Nếu **condition** có thể chuyển đổi thành true thì toán tử trả về giá trị của **exprT**; ngược lại nó trả về giá trị của **exprF**.

Hiện nay các phiên bản từ PHP 5.3+ trở về sau chúng ta có cách viết rút gọn hơn bằng toán tử elvis như sau:



Cú pháp này dịch ra văn nói sẽ là "nếu điều kiện expr1 là True thì sẽ return expr1, nếu không thì return expr2". Biểu thức bên phải của toán tử Elvis tức toán hạng thứ 2 sẽ chỉ được thực thi nếu toán hạng thứ nhất được đánh giá là False.

Chúng ta cần phải ghi nhớ thế nào được coi là False trong PHP:

* Kiểu Boolean: false
* Kiểu Integer: 0
* Kiểu Float: 0.0
* Kiểu String: '0', "0"
* String rỗng: '', ""
* Mảng rỗng: [] hoặc array()
* NULL

Chúng ta có thể sử dụng cùng lúc nhiều toán tử Elvis và nó sẽ trả về giá trị TRUE đầu tiên mà nó gặp phải

echo 0 ?: 1 ?: 2 ?: 3; // output: 1

Toán tử hợp nhất NULL ??

Từ PHP 7, toán tử hợp nhất NULL được giới thiệu và có cú pháp như sau



Nó có nghĩa là expr1 được trả về nếu expr1 tồn tại và không NULL, mặt khác biểu thức sẽ trả về expr2.

Nếu không sử dụng cú pháp ?? thì biểu thức trên sẽ được biểu diễn như sau



So sánh ?: với ??

Chúng ta sẽ làm 1 bảng so sánh 2 toán tử này với 1 biểu thức đã cho để thấy rõ sự khác biệt của chúng





4.3. Phương thức Post

4.3.1. Phương thức POST là gì?

Phương thức POST là hình thức client gửi dữ liệu lên server kèm theo dữ liệu và dữ liệu sẽ bị ẩn chứ không hiển thị trên URL như phương thức GET, vì vậy khi xây dựng form lấy thông tin từ user thì ta nên sử dụng phương thức POST vì nó bảo mật hơn.

*Ví dụ*

<!DOCTYPE html>

<html lang="en">

<head>

 <meta charset="UTF-8">

 <meta name="viewport" content="width=device-width, initial-scale=1.0">

 <title>POST</title>

</head>

<body>

 <h1>ĐĂNG NHẬP</h1>

 <form method="post" action="post.php">

 Username: <input type="text" name="username" value=""/> <br/> <br/>

 Password: <input type="post" name="password" value=""/> <br/> <br/>

 <input type="submit" name="btn" value="Đăng Nhập"/>

 </form>

</body>

</html>

Khi chúng ta chạy form này nhấn đăng nhập và xem trên URL thì sẽ thấy nó không có bổ sung một query string nào vào cả. Như vậy với phương thức POST thì dữ liệu sẽ được giấu đi.

4.3.2. Lấy dữ liệu từ phương thức POST

Để lấy dữ liệu từ phương thức POST thì ta sẽ sử dụng biến **$\_POST**, đây là một mảng chứa danh sách tất cả dữ liệu từ client gửi lên bằng phương thức POST.

Chúng ta bổ sung thêm đoạn code php sau form

<?php

 var\_dump($\_POST);

 ?>

Sau đó chạy lại ta và nhập vào user name , pass sẽ thấy



Ví dụ với phương thức POST

<!DOCTYPE html>

<html lang="en">

<head>

 <meta charset="UTF-8">

 <meta name="viewport" content="width=device-width, initial-scale=1.0">

 <title>POST2</title>

</head>

<body>

 <!-- ------------- Code HTML ------------------ -->

 <h1>ĐĂNG NHẬP</h1>

 <form method="post" action="post2.php">

 Username: <input type="text" name="username" value=""/> <br/> <br/>

 Password: <input type="post" name="password" value=""/> <br/> <br/>

 <input type="submit" name="btn" value="Đăng Nhập"/>

 </form>

 <!-- ------------- Code PHP ------------------- -->

 <?php

 if (!empty($\_POST['btn'])) // Lệnh kiểm tra nếu biến không rỗng thì chạy code

 {

 // B1: Lấy thông tin

 $username = isset($\_POST['username']) ? $\_POST['username'] : '';

 $password = isset($\_POST['password']) ? $\_POST['password'] : '';

 // B2: Kiểm tra dữ liệu

 if (!$password || !$username){

 echo 'Bạn chưa nhập đủ thông tin';

 }

 else if ($password != '12hung34' || $username != 'admin'){

 echo 'Thông tin đăng nhập bị sai';

 }

 else{

 echo 'Đăng nhập thành công!';

 }

 }

 ?>

</body>

</html>

*Lưu ý: nếu xuất hiện lỗi* **Notice**: Undefined index: btn in **F:\ltphp\post\post2.php** on line **19**

Thông báo này được hiển thị khi PHP cố gắng truy cập vào chỉ số không tồn tại của array. Dưới đây là cách giải quyết vấn đề:

Kiểm tra chỉ số có tồn tại hay không trước khi truy cập đến nó. Chúng ta có thể sử dụng isset() hoặc empty().

[4. Xử lý Form 1](#_Toc70321051)

[4.1. Giới thiệu về form trong PHP 1](#_Toc70321052)

[a. Request 1](#_Toc70321053)

[b. Response 1](#_Toc70321054)

[4.2. Phương thức Get 1](#_Toc70321055)

[Client gửi lên 1](#_Toc70321056)

[Server nhận dữ liệu 1](#_Toc70321057)

[4.2.1. Phương thức GET là gì? 1](#_Toc70321058)

[4.2.2. Lấy dữ liệu từ phương thức GET 2](#_Toc70321059)

[Ví dụ dùng phương thức GET 3](#_Toc70321060)

[Hàm date() 3](#_Toc70321061)

[Hàm isset() 4](#_Toc70321062)

[Hàm empty() 5](#_Toc70321063)

[Toán tử ternary (ba ngôi) 5](#_Toc70321064)

[Toán tử hợp nhất NULL ?? 6](#_Toc70321065)

[So sánh ?: với ?? 7](#_Toc70321066)

[4.3. Phương thức Post 8](#_Toc70321067)

[4.3.1. Phương thức POST là gì? 8](#_Toc70321068)

[4.3.2. Lấy dữ liệu từ phương thức POST 8](#_Toc70321069)

[Ví dụ với phương thức POST 9](#_Toc70321070)