CODEGYM

Raising the bar

String & Regex

Mục tiêu

- Trình bày được khái niệm chuỗi (String)
- Biết khai báo chuỗi trong PHP
- Sử dụng được toán tử nối chuỗi
- Sử dụng được các hàm thao tác với chuỗi
- Trình bày được biểu thức chính quy (Regular Expression)
- Sử dụng các hàm thao tác với biểu thức chính quy: preg_match(), preg_match_all(), preg_split(), preg_replace()

String – Chuỗi là gì?

- Là kiểu dữ liệu cơ bản trong PHP
- Là dãy liên tiếp các ký tự
- Được bao quanh bởi dấu nháy đơn (‘’) hoặc dấu nháy kép (“”)
- Chuỗi không chứa ký tự gọi là chuỗi rỗng
- Ví dụ:
  - $string_1 = "Đay la vi du mot chuoi trong PHP";
  - $string_2 = 'Chuoi su dung dau nhay don';
- Phân biệt khác nhau khi sử dụng giữa sử dụng dấu nháy đơn và dấu nháy kép

Khai báo chuỗi

- $studentName = “Nguyễn Thanh Phong”;
- $studentAddress = ‘Hà Nội’;
- $studentNotes = “Tôi vừa đọc cuốn sách: ‘Cuốn theo chiều gió’”;

Toán tử nối chuỗi

- Để nối hai hay nhiều chuỗi chúng ta sử dụng toán tử chấm (.)
  - "Code"."Gym" => "CodeGym"
  - "Gia tri cua x: ".10 => "Gia tri cua x: 10"
  - "Gia tri của x: ".$x." va Gia tri cua y: ".$y

Các hàm làm việc với chuỗi

Tên hàm	Ý nghĩa
strlen($string)	Hàm trả về độ dài chuỗi $string
str_word_count($string)	Hàm trả về số từ trong chuỗi $string
strrev($string)	Hàm đảo ngược chuỗi $string
strpos($string,$text)	Hàm tìm kiếm chuỗi $text trong chuỗi $string
str_replace($find, $replace, $string)	Hàm tìm kiếm chuỗi $find, thay thế chuỗi đó bằng $replace trong chuỗi ban đầu $string
substr($string,$start,$length)	Hàm này có tác dụng cắt chuỗi $string, bắt đầu ở vị trí $start và có giới hạn $length
trim($string, $charlist)	Hàm có tác dụng loại bỏ khoảng trắng hoặc các ký tự $charlist trong chuỗi $string

Regular Expression – Biểu thức chính quy

- Thường được gọi là Regex hay RegExp
- Là thuật toán khớp mẫu mạnh mẽ có thể được thực hiện trong một biểu thức
- Nhóm các ký tự, ký hiệu được sử dụng để tìm kiếm văn bản
- Tại sao lại sử dụng biểu thức chính quy:
  - ✓ Đơn giản hóa việc xác định các mẫu trong chuỗi dữ liệu bằng cách gọi một hàm duy nhất
  - ✓ Xác thực dữ liệu người dùng nhập vào như username, pasword...
  - ✓ Highlight từ khóa trong kết quả tìm kiếm
  - ✓ Tạo mẫu HTML tùy chỉnh
  - ✓ Tăng hiệu suất chương trình lên nhiều lần

Khai báo biểu thức chính quy

- Cú pháp:

  $pattern = '/các ký tự của BTCQ- metacharacters/flags';

- Metacharacters: Là tập các ký tự dùng trong biểu thức chính quy
- Flags: Các cờ đánh dấu
- Ví dụ:
  - ✓ $pattern = '/abc/';
  - ✓ $pattern = '/^a/';
  - ✓ $pattern = '/^[a-zA-Z0-9_]+$/';
  - ✓ $pattern = '/CodeGym/ig';

Metacharacters - 1

- Các ký tự dùng trong biểu thức chính quy được gọi là metacharacters
- Sau đây là một số các ký tự cơ bản thường sử dụng:

Metacharacter	Mô tả	Ví dụ
^	Bắt đầu của chuỗi	/^a/
$	Kết thúc của chuỗi	/a$/
*	Lặp ít nhất 0 lần	/^a*$/
+	Lặp ít nhất 1 lần	/^a+$/
?	Lặp 0 hoặc 1 lần	/^a?$/
{n,m}	Lặp n đến m lần	/^a{1,5}$
[]	Tập hợp ký tự	/^[xyz]$/

Metacharacters - 2

Metacharacter	Mô tả	Ví dụ
[^]	Tập hợp các ký tự phủ định	/^[^xyz]+$/
()	Nhóm các ký tự	/(CodeGym)/
.	Khớp với bất kỳ ký tự đơn nào trừ newline	/^a.$/
-	Khoảng liên tiếp các giá trị	/^[a-zA-Z]+$/
\	Đưa các ký tự đặc biệt	/^[0-9]{2}\-[0-9]{2}\-[0-9]{4}$/
\d	Chữ số, tương đương [0-9]	/^\d{2}\-\d{2}\-\d{4}$/
\D	Phủ định của \d, tương đương [^\d]	
\w	Chữ cái, chữ số, dấu gạch dưới, tương đương [a-zA-Z0-9_]	/^\w+$/
\W	Phủ định của \w, tương đương [^\w]	
\s	Ký tự trắng	/^[\w\s]+$/
\S	Không phải ký tự trắng, tương đương [^\s]	

Flags

Flag	Diễn tả	Ví dụ
i	Thiết lập không phân biệt chữ hoa chữ thường	/^CodeGym$/i
g	Tìm kiếm toàn chuỗi	/^CodeGym$/ig
m	So khớp trên từng dòng đối với văn bản đa dòng và có sử dụng cặp “^$”	

Các hàm thao tác với biểu thức quy tắc

- PHP xây dựng sẵn các hàm để thao tác với biểu thức chính quy
- Một số hàm thường xuyên sử dụng:
  - ✓ preg_match()
  - ✓ preg_match_all()
  - ✓ preg_split()
  - ✓ preg_replace()

Hàm preg_match()

- Cú pháp: preg_match($pattern,$subject,&$matches)
- Trong đó:
  - ✓$pattern là biểu thức Regular Expression
  - ✓$subject là chuỗi cần kiểm tra
  - ✓$matches là kết quả trả về, đây là một tham số tùy chọn, truyền vào ở dạng tham chiếu.
- Kết quả: Hàm preg_match() sẽ trả về TRUE nếu so khớp $pattern với $subject và FALSE nếu không khớp.
- Ví dụ:

```
<?php
$string = 5;
$pattern = '/^[0-9]$/';
if (preg_match($pattern, $string)) {
echo 'Khớp';
} else {
echo 'Không khớp';
}
?>
```

Hàm preg_match_all()

- Cú pháp: preg_match_all($pattern,$subject,&$matches)
- Trong đó:
  - ✓$pattern là biểu thức Regular Expression
  - ✓$subject là chuỗi cần kiểm tra
  - ✓$matches là kết quả trả về, đây là một tham số tùy chọn, truyền vào ở dạng tham chiếu.
- Kết quả: Hàm preg_match_all() tương tự như hàm preg_match(), tuy nhiên preg_match_all() sẽ trả về toàn bộ các giá trị được so khớp, thay vì preg_match() chỉ trả về giá trị đầu tiên được so khớp
- Ví dụ:

```
<?php
$subject = "Chào mừng bạn đến với CodeGym. CodeGym - Hệ thống đào tạo lập trình hiện đại.";
$pattern = '/CodeGym/';
print('<pre>');
preg_match_all($pattern, $subject, $matches);
print_r($matches);
print('</pre>');
?>
```

Hàm preg_split()

- Cú pháp: preg_split($pattern,$subject)
- Trong đó:
  - ✓$pattern là biểu thức Regular Expression
  - ✓$subject là chuỗi cần tìm kiếm và chia nhỏ
- Kết quả: Sử dụng hàm pre_split để chia nhỏ chuỗi $subject thành mảng chứa các chuỗi con dựa vào $pattern
- Ví dụ:

```
<?php
$ip = "192.168.1.1";
$iparr = preg_split ("/\./", $ip);
print "$iparr[0] <br />";
print "$iparr[1] <br />" ;
print "$iparr[2] <br />" ;
print "$iparr[3] <br />" ;
?>
```

Hàm preg_replace()

- Cú pháp: preg_replace($pattern,$replacement,$subject)
- Trong đó:
  - ✓$pattern : Biểu thức Regular Expression
  - ✓$subject : Chuỗi nhập vào để tìm kiếm và thay thế
  - ✓$replacement : Giá trị thay thế, có thể là chuỗi hoặc mảng
- Kết quả: Hàm này tìm trong $subject các chuỗi con phù hợp với mẫu $pattern, thay thế chuỗi tìm thấy bởi $replacement
- Ví dụ:

```
<?php
$str = "Vi du ve ham preg_replace 21321 878";
$str = preg_replace("/[0-9]+/", "2000", $str);
print $str;
?>
```

Tổng kết

- Dữ liệu hiển thị trên trang web đa phần dưới dạng chuỗi. Vì vậy việc xử lý chuỗi trong lập trình là rất quan trọng.
- Việc hiểu và nắm vững kiến thức xử lý chuỗi giúp tối ưu hóa hiệu suất của website và đẩy nhanh tiến độ thiết kế website
- Sử dụng thành thạo hàm thao tác với chuỗi
- Biểu thức chính quy là thuật toán khớp mẫu mạnh mẽ có thể được thực hiện trong 1 biểu thức
- Sử dụng các hàm thao tác với biểu thức chính quy: preg_match(), preg_match_all(), preg_split(), preg_replace()

Raising the bar