HỌC PHẦN "LẬP TRÌNH WEB VỚI PHP"

(Web programming with PHP)

GVGD: ThS. Trần Mạnh Đông

PHP cơ bản

NỘI DUNG (2)

- Biểu thức điều kiện
- Cấu trúc điều khiển IF
- Cấu trúc lựa chọn SWITCH-CASE

Biểu thức điều kiện

- ***Toán tử bằng*** (*equality operator*):

Toán tử	Ví dụ	Mô tả
==	$x == $y	So sánh bằng, trả về true nếu $x bằng $y
!=	$x != $y	So sánh không bằng, trả về true nếu $x không bằng $y
<>	$x <> $y	So sánh không bằng, trả về true nếu $x không bằng $y

  - Trong phép so sánh, sử dụng ***toán tử bằng*** là đủ. Tuy nhiên, khi kiểm tra phức tạp, có thể nhận được những kết quả không mong muốn khi sử dụng toán tử bằng. Do toán tử bằng thực hiện ép kiểu (chuyển dữ liệu từ kiểu này sang kiểu khác) để chuyển các dữ liệu khác kiểu sẽ chuyển về cùng kiểu trước khi so sánh được thực hiện

Biểu thức điều kiện...

- ***Toán tử bằng***:...
  - Các quy tắc ép kiểu trong PHP

Toán hạng 1	Toán hạng 2	Thao tác
NULL	Chuỗi	Chuyển đổi NULL sang chuỗi rỗng và so sánh hai chuỗi
Boolean hoặc NULL	Không phải chuỗi	Chuyển đổi cả hai về Boolean và so sánh
Chuỗi	Số	Chuyển chuỗi thành số và so sánh hai số
Chuỗi số	Chuỗi số	Chuyển cả hai thành chuỗi số và so sánh hai số
Chuỗi văn bản	Chuỗi văn bản	So sánh hai chuỗi giống như khi ta dùng hàm strcmp(string compare)

Biểu thức điều kiện…

- ***Toán tử bằng***:…
  - Do sử dụng ép kiểu, hệ quả một số trường hợp không cho ra kết quả mong muốn

Biểu thức	Kết quả	Mô tả
NULL==’’	True	NULL được chuyển về chuỗi rỗng
NULL==false	True	NULL được chuyển về FALSE
NULL==0	True	NULL sẽ bằng với bất kỳ giá trị tương ứng với FALSE như 0, “0”, 0.00
FALSE=’0’	True	Chuỗi rỗng và ‘0’ sẽ bị ép kiểu về FALSE
TRUE=’FALSE’	True	Bất cứ chuỗi nào khác rỗng đều bị ép về TRUE
3.5==’\t3.5 mi’	True	Chuỗi sẽ được chuyển về số và so sánh
INF==’INF’	False	Chuỗi ‘INF’ bị ép kiểu về 0
0==’’	True	Chuỗi rỗng bị ép về 0
0==’harris’	True	Bất kỳ chuỗi nào không chứa số đều bị ép về 0

Biểu thức điều kiện…

- ***Toán tử đồng nhất*** (*identity operator*): không thực hiện ép kiểu. Khi đó, nếu giá trị khác kiểu được so sánh, sẽ trả về luôn FALSE

Toán tử	Ví dụ	Mô tả
===	$x === $y	So sánh loại và giá trị, trả về true nếu $x cùng loại và cùng giá trị với $y
!==	$x !== $y	So sánh không cùng loại, trả về true nếu $x không cùng loại với $y

- *Lưu ý*: Khi sử dụng toán tử bằng, tốt nhất nên chuyển về cùng kiểu dữ liệu trước khi so sánh.

Biểu thức điều kiện…

- ***Toán tử quan hệ*** (*relation operator*): cũng giống như toán tử bằng, toán tử quan hệ cũng thực hiện việc ép kiểu khi hai biến không cùng kiểu để thực hiện so sánh

Toán tử	Ví dụ	Mô tả
>	$x > $y	So sánh lớn hơn, trả về true nếu $x lớn hơn $y
<	$x < $y	So sánh bé hơn, trả về true nếu $x bé hơn $y
>=	$x >= $y	So sánh lớn hơn hoặc bằng, trả về true nếu $x lớn hơn hoặc bằng $y
<=	$x <= $y	So sánh bé hơn hoặc bằng, trả về true nếu $x bé hơn hoặc bằng $y

Biểu thức điều kiện…

- ***Toán tử quan hệ***:..
  - So sánh chuỗi và số bằng toán tử quan hệ

Biểu thức	Kết quả	Mô tả
1 <’5’	True	Chuỗi “5” được chuyển thành số 5
‘10’ < 5	false	Chuỗi 10 được chuyển thành số 10

  - So sánh các chuỗi bằng toán tử quan hệ

Biểu thức	Kết quả	Mô tả
‘apple’<’orange’	True	Chữ trước nhỏ hơn chữ sau
‘apple’<’appletree’	True	Khi các ký tự tương tự nhau, chuỗi ngắn hơn sẽ nhỏ hơn chuỗi dài hơn
‘Orange’<’apple’	True	Chữ viết hoa sẽ nhỏ hơn chữ in thường
‘@’<’$’	False	Các ký tự khác được so sánh dựa trên giá trị ASCII của nó

Biểu thức điều kiện…

- ***Toán tử quan hệ***:…
  - Một số trường hợp cho kết quả bất thường khi sử dụng

Biểu thức	Kết quả	Mô tả
0 <= ‘test’	True	Chuỗi ‘test’ được chuyển thành 0
‘ ’ < 1	True	Chuỗi rỗng được chuyển thành 0
false<true	True	FALSE được coi là nhỏ hơn TRUE
null < true	True	NULL được chuyển thành FALSE

Biểu thức điều kiện...

- ***Toán tử logic*** (*logical operator*):

Toán tử	Ví dụ	Mô tả
and	$x and $y	Trả về giá trị true nếu cả $x và $y đều đúng
&&	$x && $y	Trả về giá trị true nếu cả $x và $y đều đúng
or	$x or $y	Trả về giá trị true nếu cả $x hoặc $y đều đúng
||	$x || $y	Trả về giá trị true nếu cả $x hoặc $y đều đúng
xor	$x xor $y	Trả về giá trị true nếu một trong hai $x hoặc $y đúng, nhưng không phải cả 2 đều đúng
!	!$x	Trả về giá trị true nếu $x không đúng

Biểu thức điều kiện...

- Thứ tự ưu tiên của các toán tử trong biểu thức điều kiện

Thứ tự	Toán tử	Mô tả
1	!	Toán tử NOT
2	<, <=, >, >=, <>	Toán tử quan hệ
3	==,!=,===,!==	Toán tử bằng và toán tử đồng nhất
4	&&	Toán tử AND
5	||	Toán tử OR

Biểu thức điều kiện…

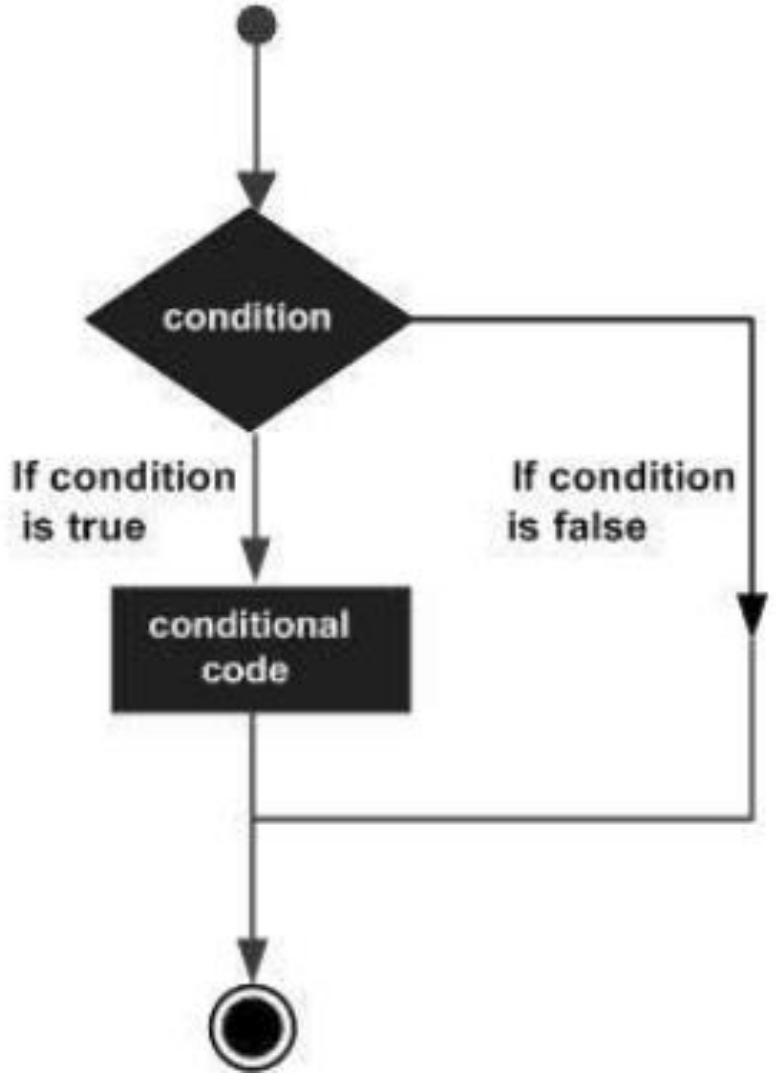
- **Điều kiện** là một biểu thức logic có thể trả về giá trị TRUE hoặc FALSE.
- **Điều kiện** thường được sử dụng để kiểm tra một điều kiện nào đó và thực hiện một hành động tương ứng.
- PHP hỗ trợ nhiều cấu trúc điều kiện khác nhau, chẳng hạn như:
  - *Cấu trúc rẽ nhánh IF, IF-ELSE*
  - *Cấu trúc lựa chọn SWITCH-CASE*

Cấu trúc điều khiển IF

- Cú pháp

```
if(điều kiện)
{
    khối lệnh
}
```

```
if(điều kiện)
{
    khối lệnh 1
}
Else
{
    khối lệnh 2
}
```



Cấu trúc điều khiển IF…

- Nếu sau lệnh IF và mệnh đề ELSE chỉ có một câu lệnh: Không cần thêm ngoặc nhọn bao lệnh này
- Có thể lồng lệnh IF trong mệnh đề IF, ELSE IF hoặc ELSE của câu lệnh IF khác
  - Lệnh có thể lồng sâu bao nhiêu tầng tùy ý
  - Sử dụng ngoặc nhọn và thụt lề hợp lý sẽ giúp đoạn mã dễ đọc hơn

Cấu trúc điều khiển IF…

- Toán tử **?**
  - Cú pháp:

```
(biểu thức điều kiện)?<kết quả khi điều kiện đúng>:<kết quả khi điều kiện sai>
```

  - Ý nghĩa: dùng để thay thế cho cấu trúc điều khiển if…else với một câu lệnh bên trong
  - Có thể lồng nhiều toán tử ?: với nhau

Cấu trúc điều khiển IF…

- **Dạng: if … else if … else**

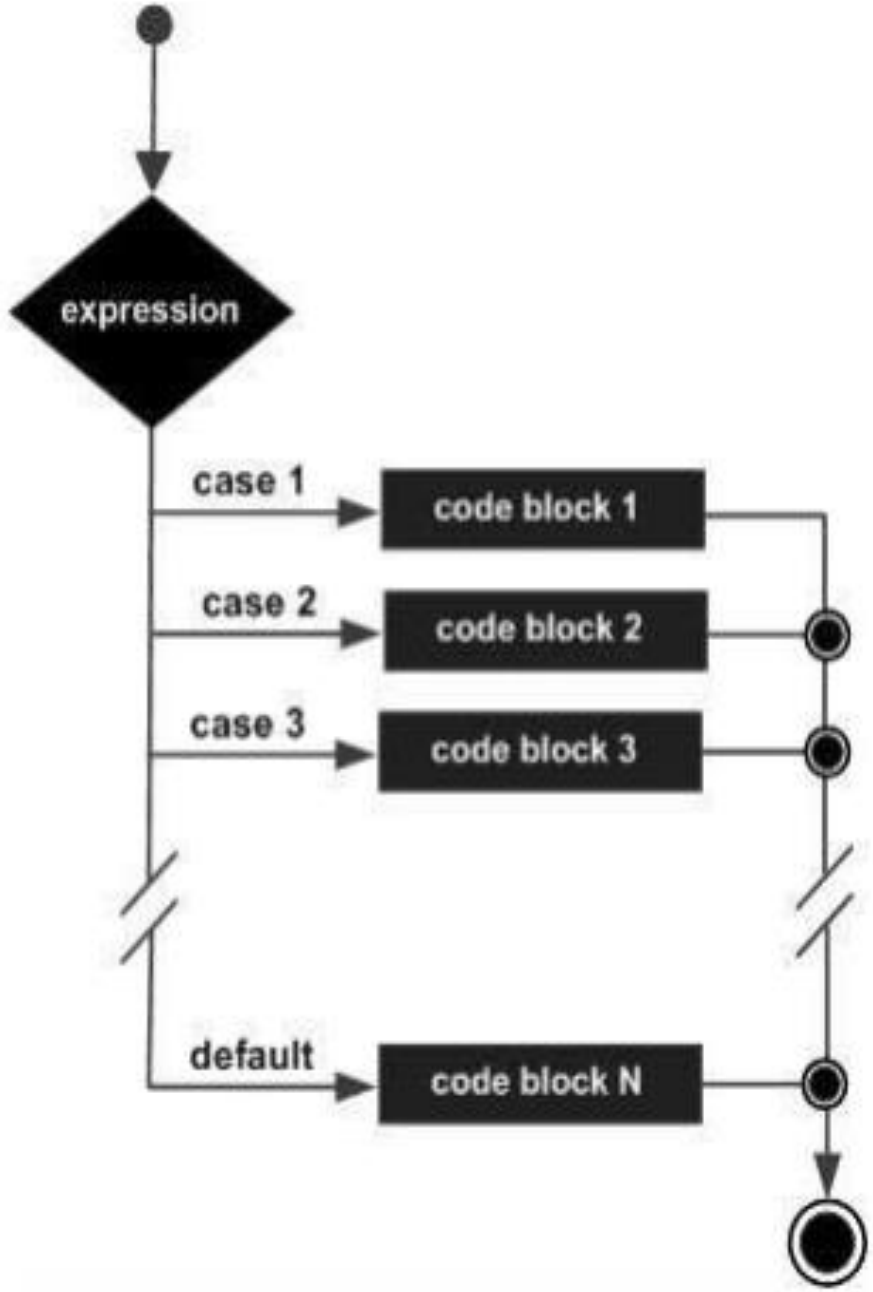
```
if(điều kiện 1)
{
    khối lệnh 1
}
else if(điều kiện 2)
{
    khối lệnh 2
}
…
Else
{
  khối lệnh khi không thỏa mãn các điều kiện trên
}
```

- *Lưu ý*: Khi viết mệnh đề ELSE IF: có thể viết riêng rẽ hai từ khóa ELSE và IF (có khoảng trắng ở giữa) hoặc có thể viết từ khóa liền ELSEIF (không khoảng trắng)

Cấu trúc lựa chọn SWITCH

- Cú pháp

```
switch(biến điều kiện)
{
        case giá trị 1:
                khối lệnh 1
                break;
        case giá trị 2:
                khối lệnh 2
                break;
        ...
        [default: khối lệnh khi không thỏa tất cả các
case trên]
}
```



Cấu trúc lựa chọn SWITCH…

- Câu lệnh SWITCH bắt đầu bằng việc định trị biểu thức SWITCH trong ngoặc đơn
- Sau khi định trị biểu thức, câu lệnh SWITCH sẽ chuyển điều khiển sang nhãn CASE có giá trị bằng với giá trị của biểu thức. Tiếp đó, nó thực thi khối lệnh trong trường hợp này và chỉ dừng lại khi gặp lệnh break hoặc đến cuối lệnh SWITCH
- Trường hợp default là không bắt buộc và có thể bỏ qua. Nếu thêm vào, chỉ có thể có duy nhất một trường hợp mặc định. Nó thường là trường hợp cuối cùng trong lệnh, tuy nhiên có thể đặt ở bất kỳ đâu
- Giá trị trong nhãn CASE có thể là giá trị nguyên bản hoặc một biểu thức
- Nếu CASE không chứa lệnh break, đoạn mã sẽ tuần tự thực thi các lệnh trong nhãn tiếp theo
- Có thể lồng lệnh IF hoặc SWITCH vào nhãn CASE của lệnh SWITCH

Cấu trúc lựa chọn SWITCH…

- Ví dụ:

```php
<?php
   //Ví dụ 2: Nhập tháng tìm số ngày
   $month = 2;

   switch ($month) {
      case 2:
         echo "Tháng có 28 ngày (năm không nhuận) và 29 ngày (năm nhuận)";
         break;
      case 4:
      case 9:
      case 11:
         echo "Tháng có 30 ngày";
         break;
      case 1:
      case 3:
      case 5:
      case 7:
      case 8:
      case 10:
      case 12:
         echo "Tháng có 31 ngày";
         break;
      default:
         echo "Tháng không hợp lệ";
   }
?>
```

Demo

- Biểu thức điều kiện
- Cấu trúc IF
- Cấu trúc SWITCH

BTVN- Lab 1