VIDEO 1.15

A blue screen with white text

Description automatically generated



Một tập lệnh PHP được tạo thành từ một loạt các câu lệnh, trong đó câu lệnh có thể là một phép gán, một hàm được gọi là một vòng lặp, các điều kiện khác, vv... Như bạn đã biết, trong PHP, câu lệnh được kết thúc bằng dấu chấm phẩy, và bạn có thể nhóm nhiều câu lệnh lại bằng cặp ngoặc nhọn.

Cấu trúc điều khiển là một cơ chế cho phép bạn nhóm nhiều câu lệnh lại và cũng cho phép bạn kiểm soát luồng thực thi của mã.

Ví dụ, bạn chỉ có thể thực thi A nếu điều kiện B được thoả mãn hoặc tiếp tục thực thi A cho đến khi điều kiện B được đáp ứng.

Cấu trúc điều khiển đầu tiên chúng ta sẽ tìm hiểu là câu lệnh điều kiện if. Cú pháp là if theo sau là một điều kiện nào đó, sau đó là cặp ngoặc nhọn và bên trong ngoặc nhọn bạn đặt các khối mã hoặc câu lệnh mà bạn muốn thực thi chỉ khi điều kiện đúng. Điều kiện này có thể là bất kỳ loại biểu thức nào. Nó có thể đơn giản chỉ là một biến như ví dụ sau

A screenshot of a computer

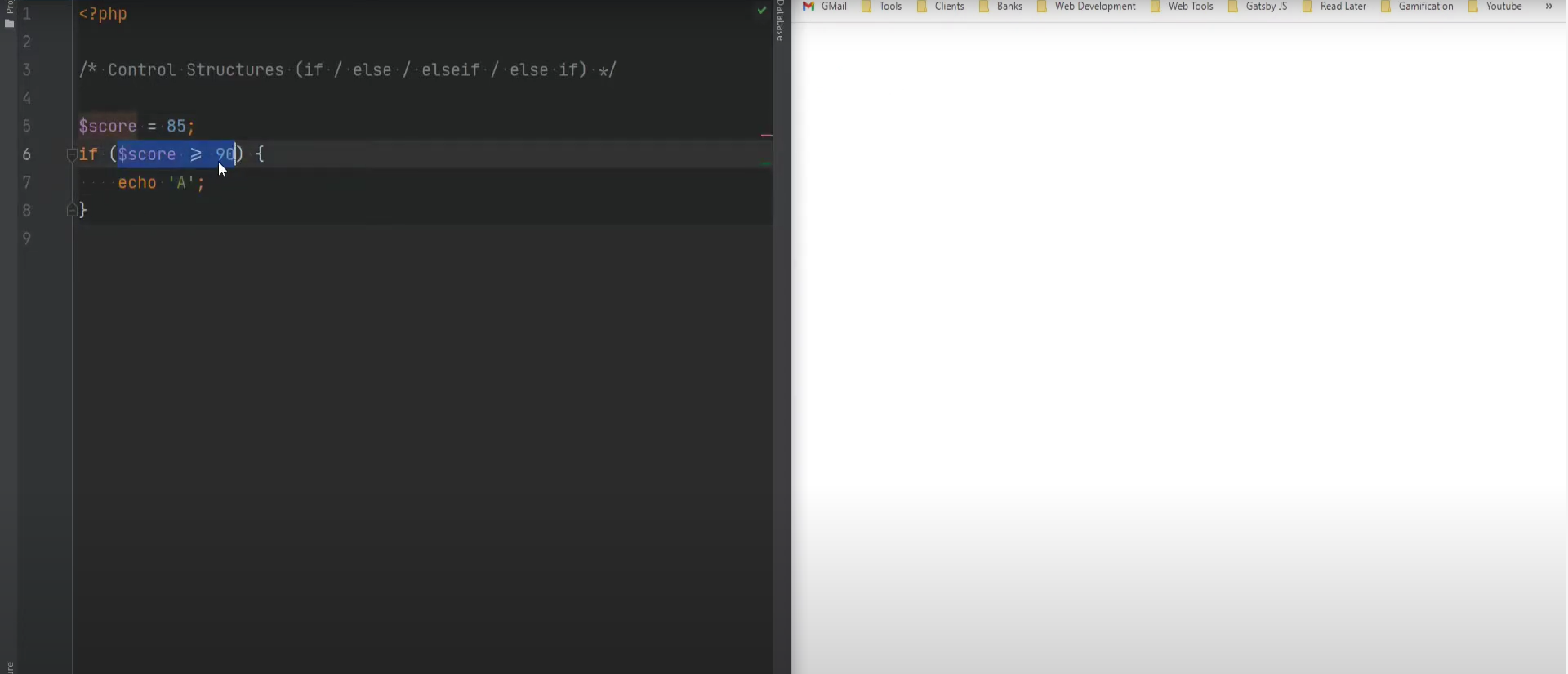
Description automatically generated

Hoặc nó có thể phức tạp hơn bằng cách sử dụng nhiều toán tử, bao gồm toán tử logic và vân vân.

Ví dụ, hãy thực hiện một hệ thống chấm điểm đơn giản. Giả sử điểm là 95 và chúng ta có thể sử dụng if “score >= 90”, echo "A". A screenshot of a computer

Description automatically generated

Nếu chúng ta làm mới trang, chúng ta sẽ nhận được A. Nếu chúng ta thay đổi thành 85, kết quả sẽ không được in ra.



Như bạn có thể thấy, câu lệnh “echo = A” chỉ được thực thi nếu điều kiện “$score≥ 90” đúng.

Đối với các câu lệnh đơn giản như vậy, khi bạn có một dòng, bạn có thể loại bỏ cặp ngoặc nhọn và câu lệnh vẫn hoạt động.

A screenshot of a computer

Description automatically generated

Vì vậy nếu chúng tôi làm mới thì không có gì được in ra

Nếu chúng ta đổi lại thành 95, lại in ra A.

A screenshot of a computer

Description automatically generated

Theo cá nhân tôi khuyến nghị luôn sử dụng cặp ngoặc nhọn vì nó làm code dễ đọc hơn.

Vậy điều gì xảy ra khi điều kiện thất bại?  
Bạn có thể làm như sau: nếu điều kiện thất bại, bạn có thể thực thi một khối mã mặc định.Và đó chính là sử dụng câu lệnh “else” ở đây

A screenshot of a computer

Description automatically generated  
Vì vậy, bạn có thể sử dụng else và bạn đặt khối mã mà bạn muốn thực thi trong điều kiện khi biểu thức “$score ≥ 95” không đánh giá là đúng trong dấu ngoặc nhọn.  
Vì vậy, bạn có thể nói “echo F” trong câu lệnh “else” .Nếu làm mới, chúng ta đang nhận được A: A screenshot of a computer

Description automatically generatedNhưng nếu chúng ta thay đổi thành 85, chúng ta đang nhận được F. A screenshot of a computer

Description automatically generated

Điều đó khá nghiêm khắc, phải không? Bởi vì nếu bạn có điểm số là 85, bạn sẽ không bao giờ nhận được F. Điều này có nghĩa là bạn thực sự cần nhiều điều kiện. Và bạn có thể làm điều đó bằng cách sử dụng “elseif”. Vì vậy, chúng ta có thể thực hiện “elseif ($score ≥80){echo B})”. Và sau đó, chúng ta làm như vậy cho C và D.A screen shot of a computer program

Description automatically generated

Như bạn có thể thấy, bạn có thể thay đổi nhiều điều kiện elseif nếu bạn cần. Trong trường hợp này, vì điểm là 85, chúng ta sẽ nhận được B.

A screenshot of a computer

Description automatically generated

 Nếu chúng ta thay đổi “$score = 65”, chúng ta sẽ nhận được D. Và nếu chúng ta thay đổi “$score = 65”, chúng ta sẽ nhận được C. Cuối cùng, nếu chúng ta đặt thành một giá trị nào đó như “$score = 50”, chúng ta sẽ nhận được F.

Cách khác để sử dụng “else if” là bằng cách sử dụng khoảng trắng và điều này sẽ hoạt động chính xác như cách sử dụng “elseif”

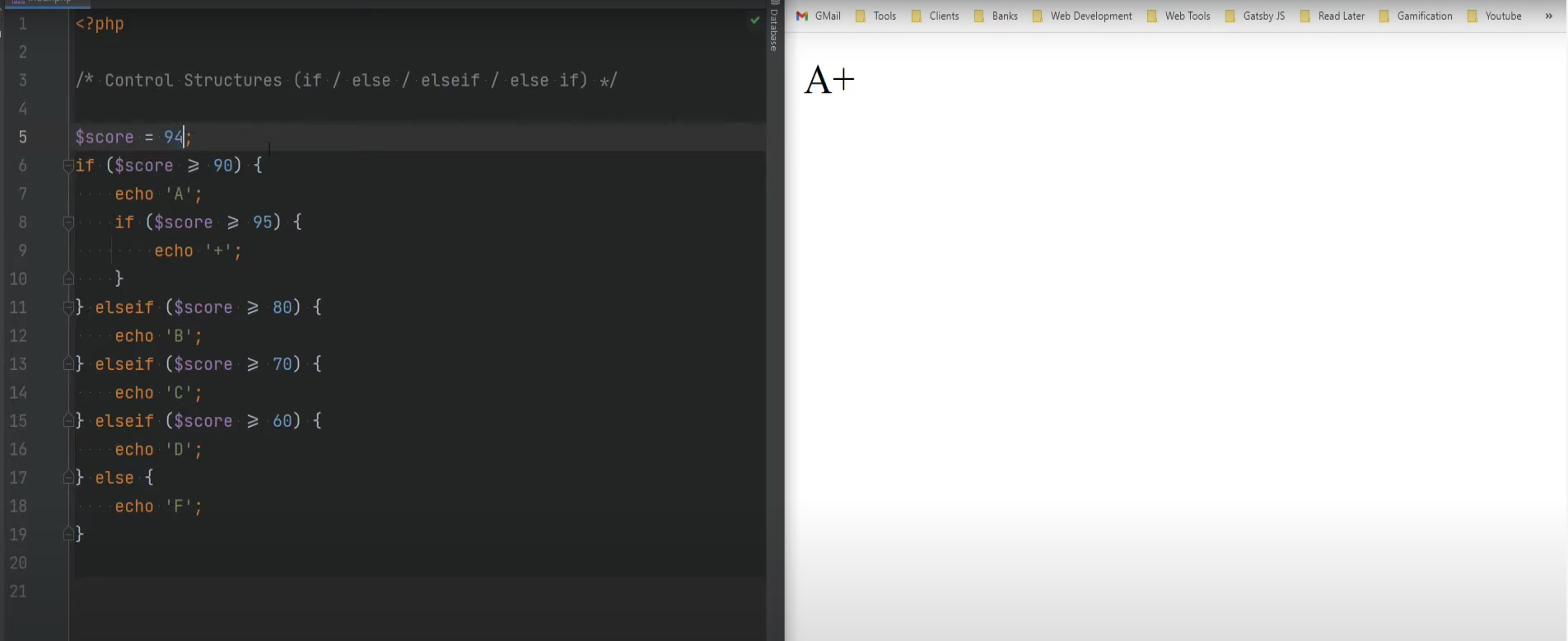
A screenshot of a computer

Description automatically generated Có một sự khác biệt cú pháp giữa cả hai nhưng khi bạn sử dụng cú pháp thay thế, mà tôi sẽ nói sau trong một phút trong HTML, bạn cần sử dụng cú pháp không có khoảng trắng “elseif”. Cá nhân tôi luôn sử dụng loại “elseif”.

Bạn cũng có thể lồng các điều kiện if else trong các điều kiện khác theo ý muốn. Như vậy, bạn có thể nói nếu điểm số lớn hơn hoặc bằng 95, echo plus. A screenshot of a computer

Description automatically generated

Vì vậy, nếu chúng ta đặt “$score =95”, chúng ta sẽ nhận được một dấu cộng A+:



Nếu chúng ta đặt “$score= 94”, chúng ta nhận được một điểm A và như vậy cho các trường hợp khác bạn có thể lồng nó trong elsif và bạn cũng có thể lồng nó trong các trường hợp khác. Như bạn có thể thấy, chúng ta đang lặp lại rất nhiều mã if else ở đây, và đừng lo lắng về điều đó. Ngay bây giờ chúng ta sẽ bàn về các phương thức tốt nhất và cách tránh việc lặp lại mã trong các video sau đó. Vì vậy, nếu chúng ta thay đổi điều này thành 76, chúng ta sẽ nhận được C+ nếu chúng ta thay đổi nó thành 74, chúng ta chỉ nhận được C.

A screenshot of a computer

Description automatically generated

Vì vậy, hãy nhúng điều này vào trong HTML và xem cú pháp thay thế. Nhưng dường như điều này trông xấu xí, phải không?

A screenshot of a computer

Description automatically generatedVì chúng ta đang nhúng rất nhiều PHP vào HTML. Ngoài ra, nếu chúng ta muốn in ra một số kiểu HTML hoặc in ra một số phần tử HTML với các giá trị của chúng ta, chúng ta có thể in HTML trực tiếp như thế này.

A screenshot of a computer

Description automatically generated

Nhưng điều đó cũng không trông dễ đọc. Hãy nhớ, hãy xóa các câu lệnh điều kiện if thêm này A computer screen with text

Description automatically generated để có ít mã hơn để làm việc.

A screenshot of a computer

Description automatically generated

Chúng ta có thể làm như sau: đóng thẻ PHP ở đây và sau đó xóa các câu lệnh if bổ sung này ra khỏi đây và chỉ in HTML và sau đó mở thẻ PHP ở đây.

A screenshot of a computer

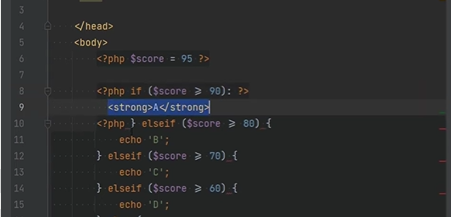
Description automatically generated

Và điều này cũng hoạt động tốt nhưng điều này cũng không dễ đọc. Và chúng ta sẽ phải làm điều tương tự cho các điều kiện elseif còn lại và điều kiện khác ở đây. May mắn là PHP cung cấp cú pháp thay thế. Và để tôi chỉ cho bạn cú pháp đó là gì. Vì vậy, hãy đóng phần trên ở đây để chúng ta có điểm số 95 được xác định

A screenshot of a computer

Description automatically generated

tiếp theo chúng ta có thể mở thẻ PHP ở đây. Và cú pháp thay thế là thay vì dấu ngoặc nhọn, bạn sử dụng hai chấm và sau đó đóng thẻ PHP, và sau đó, bạn in HTML của mình:



Sau đó, bạn mở thẻ PHP một lần nữa, và đối với elseif, chúng ta cũng sử dụng hai chấm ở cuối và xóa dấu ngoặc nhọn và hãy làm điều tương tự với các đối tượng elseif khác.

A screen shot of a computer

Description automatically generated

Và cuối cùng, chúng ta cần một “endif” để đóng các câu lệnh if ban đầu, vì vậy chúng ta sẽ làm như vậy.

A screenshot of a computer

Description automatically generated

Và bây giờ như bạn có thể thấy, điều này trông sạch hơn rất nhiều. Và nếu chúng ta làm mới trang, nó hoạt động như mong đợi. Chúng ta cũng có thể sử dụng các phần tử HTML khác, chúng ta có thể thêm một số kiểu dáng, chúng ta có thể dùng “style = “color: green;””. Và sau đó, chúng ta có thể thêm “style = “color: red;”” ở đây và đặt giá trị “$score= 55”. Và nó hoạt động như mong đợi.

A screenshot of a computer

Description automatically generated

Lưu ý ở đây là khi sử dụng elseif, bạn không thể sử dụng loại có khoảng trắng “else if”, nó sẽ cho bạn một lỗi cú pháp:

A screenshot of a computer

Description automatically generated

Bạn phải sử dụng loại không có khoảng trắng “elseif”. Đây là video về điều này, bạn hiện đã biết điều kiện if else if và điều kiện else và chúng ta sẽ chuyển sang các vòng lặp trong video tiếp theo. Vì vậy, cảm ơn bạn đã xem, hãy nhấn like và đăng ký. Và tôi sẽ gặp bạn trong video tiếp theo.

VIDEO 1.16

* Vòng lặp While

Tương tự như các ngôn ngữ lập trình khác,vòng lặp trong php được sử dụng để thực thi các câu lệnh trong nhiều lần. PHP hỗ trợ các vòng lặp sau: vòng lặp while, vòng lặp do while, vòng lặp for và vòng lặp foreach.

A screenshot of a computer

Description automatically generated

Trước tiên hãy xem qua vòng lặp while vì vậy tôi sẽ tạo một bộ đếm đơn giản và in các chữ số lên tới 15. Cho biến “$i = 0” sau đó khi i ≤ 15, lặp lại i và tăng giá trị, nếu tôi làm mới trang, chúng ta sẽ nhận được các chữ số từ 0,1,...,15.A screenshot of a computer

Description automatically generated

Như bạn có thể thấy cú pháp của vòng lặp while tương tự như câu lệnh điều kiện if mà chúng ta đã trình bày trong video trước. Bạn bắt đầu bằng từ khóa while,theo sau là dấu ngoặc đơn chứa biểu thức điều kiện “while(biểu thức điều kiện){câu lệnh}”.Và miễn là biểu thức điều kiện được đánh giá là đúng sẽ tiếp tục chạy các câu lệnh chứa trong dấu ngoặc nhọn, biểu thức điều kiện này được đánh giá ở đầu mỗi lần lặp và ngay khi nó được đánh giá là sai thì vòng lặp sẽ dừng. Nếu biểu thức này được đánh giá là sai ngay lần đầu tiên sau đó vòng lặp thực sự sẽ không bao giờ chạy, vì vậy nếu tôi đặt i= 20 và làm mới trang, chúng tôi sẽ không nhận được gì.

A screenshot of a computer

Description automatically generated

Nếu tôi quên tăng biến i ở đây điều này sẽ dẫn đến một thứ gọi là vòng lặp vô hạn và vòng lặp vô hạn là thứ sẽ liên tục chạy các câu lệnh của bạn bởi vì biểu thức sẽ luôn đánh giá là đúng. Trong trường hợp này nên chúng tôi không sửa đổi biến i và i luôn bằng 0 và biểu thức này sẽ luôn đúng và điều này thực sự có thể gây hại cho ứng dụng của bạn mặc dù có một số trường hợp sử dụng cho vòng lặp vô hạn và bạn có thể đã thấy ở đâu đó trước đây, một điều gì đó ví dụ như câu lệnh “while (true)” đang được sử dụng: một trong những trường hợp sử dụng cho vòng lặp vô hạn là khi bạn muốn đợi điều gì đó xảy ra và sau đó thoát ra khỏi vòng lặp có điều kiện bằng cách sử dụng câu lệnh “break”. Ví dụ như ở đây chúng ta có thể làm gì nếu i≥15 , break và sau đó chúng ta cũng cần tăng biến i và nếu làm mới trang, chúng ta sẽ nhận được các chữ số từ 0 đến 14.

A screenshot of a computer

Description automatically generated

Câu lệnh break chấp nhận một đối số tùy chọn và đối số đó theo mặc định bằng 1 hoặc chuyển ta có chuyển một đối số số khác ví dụ như 2, dẫn đến thoát khỏi nhiều vòng lặp lồng nhau.

A screenshot of a computer

Description automatically generated

Trong trường hợp này thì điều đó không có ý nghĩa gì do chỉ có một vòng lặp duy nhất nhưng nếu chúng ta có 1 vòng lặp lồng nhau. Ví dụ ở đây while($i >10){break 2;}”.

A screenshot of a computer

Description automatically generated

Ở đây bất cứ khi nào i vượt quá 10, ta sẽ thoát ra khỏi hai cấp độ vòng lặp và trong trường hợp này là vòng lặp này. Và trong trường hợp sử dụng đối số mặc định, là 1, thì chỉ một vòng lặp bị phá vỡ và vẫn còn lại một vòng lặp vô hạn. Vì vậy, nếu làm mới trang, ta chỉ nhận được các số lên tới 10.

A screenshot of a computer

Description automatically generated

Bạn cũng có thể bỏ qua phép lặp vòng lặp hiện tại bằng cách sử dụng câu lệnh “continue” và chuyển sang lần lặp tiếp theo,ví dụ: nếu chúng ta thay đổi giá trị “true” thay bằng “$i ≤15” và xóa câu lệnh break, điều kiện nếu i là số chẵn thì hãy thục hiện câu lẹnh continue.

A close up of a blue line

Description automatically generated

Điều này sẽ trở thành 1 vòng lặp vô hạn ở đây vì ban đầu i=0 và 0 là số chẵn và do đó nó sẽ bỏ qua lần lặp continue và chuyển sang lần lặp tiếp theo. Trong trường hợp này mặc dù ta không tăng i vì ta đang tăng i xuống bên dưới khối lệnh if và continue bỏ qua mọi thứ bên dưới nó. . Vì vậy, ta sẽ chuyển sang lần lặp tiếp theo và i lại bằng 0, lặp đi lặp lại dẫn đến một vòng lặp vô hạn. Để khắc phục điều này, ta cần tăng i trước câu điều kiện if hoặc có thể tăng i trước continue. Nếu làm mới trang kết quả chỉ nhận được số lẻ:A screenshot of a computer

Description automatically generated

Để thấy rõ ta có thể thêm dấu phẩy để phân biệt các con số

A screenshot of a computer

Description automatically generated

Tương tự như câu lệnh break, continue cũng chấp nhận một đối số tùy chọn cho phép bạn tiếp tục nhiều vòng lặp lồng nhau. PHP cũng có cú pháp thay thế cho vòng lặp while tương tự như các điều kiện if chủ yếu được sử dụng khi nhúng PHP vào HTML. Thay vì mở dấu ngoặc nhọn mà bạn có thể sử dụng dấu hai chấm “:” và thay vì dấu ngoặc nhọn đóng bạn có thể dùng “endwhile;” :

A screenshot of a computer

Description automatically generated

* Vòng lặp Do-while

A screen shot of a computer

Description automatically generated

Vòng lặp do while tương tự như vòng lặp while với điểm khác biệt chính là vòng lặp do-while đảm bảo rằng các câu lệnh trong vòng lặp sẽ được thực thi ít nhất một lần bởi vì biểu thức điều kiện sẽ được kiểm tra ở cuối một lần lặp thay vì đầu một lần lặp như bạn nhớ trong vòng lặp while, nếu biểu thức điều kiện được đánh giá là sai ngay từ đầu thì vòng lặp sẽ không bao giờ chạy.

Tuy nhiên, trong vòng lặp do while, câu lệnh “echo $i++;” sẽ được chạy trước tiên và sau đó kiểm tra biểu thức điều kiện và quyết định xem có cần chạy lại hay không.

Trong trường hợp này, chúng ta sẽ nhận được các chữ số từ 0 đến 15, nếu chúng ta thay đổi giá trị “$i = 25” thì điều này sẽ đánh giá thành sai nhưng câu lệnh echo vẫn sẽ thực thi một lần

A screenshot of a computer

Description automatically generated

* Vòng lặp For

Vòng lặp for phức tạp hơn một chút so với vòng lặp while hoặc do-while, cần có ba biểu thức được phân tách bằng dấu chấm phẩy

Biểu thức đầu tiên “$i = 0” biểu thức này chỉ đánh giá lần đầu tiên

A screenshot of a computer

Description automatically generated

Biểu thức thứ hai “$i<i” là biểu thức điều kiện và nó được đánh giá ở đầu mỗi lần lặp

A screenshot of a computer

Description automatically generated

Và cuối cùng biểu thức thứ ba “$i++” được đánh giá ở cuối mỗi lần lặp để bạn có thể thấy ở phía bên phải

A screenshot of a computer

Description automatically generated

Chúng tôi có các chữ số được in từ 0 đến 14 vì ở đây chúng tôi đang tạo một biến i và gán “i= 0” thì chúng ta sẽ kiểm tra xem i có nhỏ hơn 15 hay không và việc tăng i. Mỗi biểu thức này có thể trống, chúng không cần thiết. Vì vậy, ví dụ: chúng ta có thể đặt nó như thế này

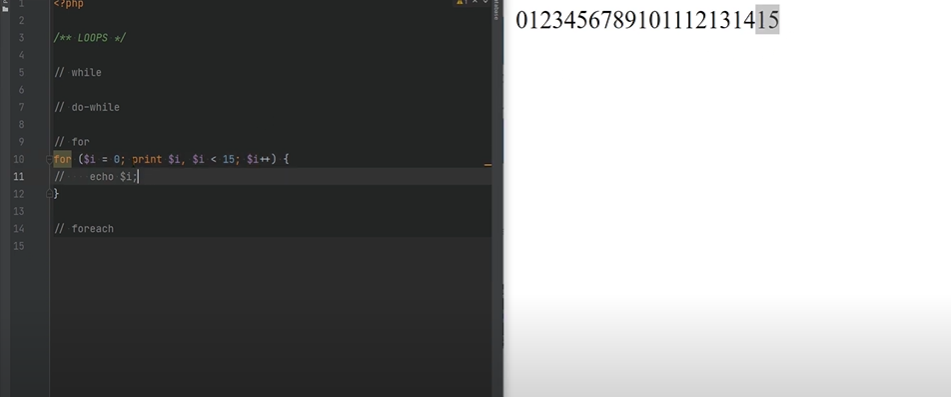
A screenshot of a computer

Description automatically generated

Và điều này sẽ hoạt động tốt về cơ bản giống như while true, nó chỉ là một vòng lặp vô hạn, chúng cũng có thể chứa nhiều biểu thức được phân tách bằng dấu phẩy để bạn có thể làm điều gì đó như “for($i=0, $i<15, print $i, $i++)” để chúng ta có thể nhận xét echo i này và nếu chúng ta làm mới trang, chúng ta sẽ nhận được chính xác cùng một kết quả A screenshot of a computer

Description automatically generated

Nếu bạn có nhiều biểu thức trong biểu thức điều kiện ở đây việc đánh giá xem vòng lặp có tiếp tục hay không sẽ được xác định bởi biểu thức cuối cùng. Vì vậy, ví dụ: chúng ta có thể “for($i=0,print $i, $i<15, $i++)”, nếu chúng ta làm mới trang thông báo rằng trước khi số cuối cùng là 14 và bây giờ số cuối cùng là 15.



Lý do là vì biểu thức điều kiện được chạy cho mỗi lần lặp và lần lặp cuối cùng nhỏ hơn 15 đang được kiểm tra sau khi chúng tôi in. Vòng lặp for cũng có thể được sử dụng để lặp các chuỗi và in ra từng ký tự hoặc lặp qua các mảng. Hãy để tôi đưa ra một ví dụ vì vậy giả sử chúng ta có “$stext = ‘hello world’” thì ở vòng lặp for chúng ta có thể thực hiện “for ($i = 0; $i < strlen($text); i++)” và in từng ký tự trên dòng riêng của nó “echo $text[$i] .<br/>.” nếu chúng ta làm mới trang, chúng ta sẽ nhận được mỗi ký tự trên một dòng riêngA screenshot of a computer

Description automatically generated

Nếu chúng ta thay đổi ký tự này thành một mảng thì chúng ta có thể thay đổi độ dài chuỗi thành count vì count trả về độ dài của một mảng và điều này có thể vẫn giữ nguyên và nếu chúng tôi làm mới, chúng tôi sẽ in ra các phần tử của một mảng.

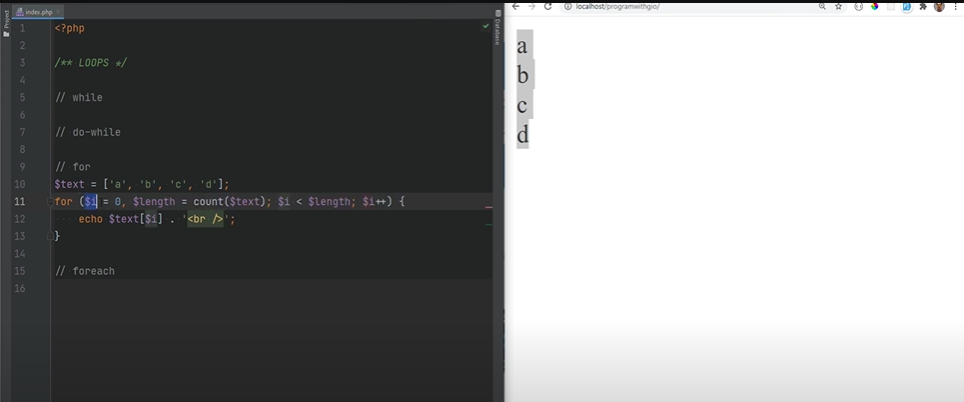
A screenshot of a computer

Description automatically generated

Tất nhiên bạn có thể làm điều tương tự bằng cách sử dụng vòng lặp do while hoặc while.

A screenshot of a computer

Description automatically generated

Bây giờ, điều này có vấn đề về hiệu suất ở đây vì bạn biết các biểu thức điều kiện chạy ở đầu mỗi lần lặp vì vậy chúng tôi đang gọi hàm count ở đây cho mỗi lần lặp trong trường hợp này chúng tôi chỉ gọi nó bốn lần, đây không phải là vấn đề lớn nhưng trong trường hợp mảng của bạn có thể quá lớn hoặc lệnh gọi hàm của bạn có thể tốn kém thì điều này có thể. Theo tôi, gây ra một số vấn đề về hiệu suất,tốt nhất là không thực hiện bất kỳ lệnh gọi hàm không cần thiết nào, vì vậy thay vì làm điều này, điều bạn có thể làm là bạn có thể tạo một biến có tên là length và gán biến đó trong biểu thức đầu tiên của vòng lặp để bạn có thể làm độ dài bằng để đếm văn bản 

Và như bạn biết biểu thức đầu tiên chỉ chạy lần đầu tiên nên số đếm sẽ chỉ được gọi một lần nếu chúng ta làm mới, chúng ta sẽ nhận được điều tương tự

A screenshot of a computer

Description automatically generated

Một tùy chọn khác là thay vì đặt nó trong vòng lặp for, bạn có thể chỉ cần đặt nó phía trên vòng lặp for, và nếu làm mới, chúng tôi nhận được cùng một lưu ý đầu ra rằng vấn đề về hiệu suất không chỉ áp dụng cho vòng lặp for mà bạn có thể gặp vấn đề về hiệu suất tương tự khi sử dụng vòng lặp while hoặc do while. A screenshot of a computer

Description automatically generated

* Vòng lặp for each

Vòng lặp for each cho mỗi vòng lặp được sử dụng để lặp qua các mảng hoặc đối tượng, Ví dụ:hãy tạo một mảng các ngôn ngữ lập trình và in từng ngôn ngữ trên dòng riêng của nó để cú pháp dành cho mỗi ngôn ngữ và trong dấu ngoặc đơn, chúng ta đặt biểu thức có thể lặp lại và trong trong trường hợp này nó chỉ là một biến mảng nên chúng ta sẽ đặt các ngôn ngữ lập trình sau đó chúng ta sử dụng as và sau đó là một loại biến nào đó nên trong trường hợp này hãy gọi nó là ngôn ngữ và sau đó chúng ta có thể lặp lại ngôn ngữ vàsau đó một dòng mới chúng ta làm mới trang

A screenshot of a computer

Description automatically generated

Và tôi nhận từng ngôn ngữ lập trình trên dòng riêng của nó, vì vậy điều đang xảy ra ở đây là trong mỗi lần lặp, nó sẽ gán giá trị của phần tử hiện tại trong một mảng cho ngôn ngữ biến đổi nếu bạn cố gắng sử dụng foreach trên thứ gì đó không phải là mảng hoặc không phải là một mảng đối tượng, nó sẽ báo lỗi cho bạn.

Ví dụ: nếu tôi đặt một số dữ liệu vô hướng “foreach(‘test’ as $language)” và tôi làm mới, tôi sẽ nhận được cảnh báo

A screenshot of a computer

Description automatically generated

Bạn cũng có thể truy cập các khóa của mảng bằng cách sử dụng biến khóa ở đây và sau đó là dấu bằng và dấu lớn hơn và tất nhiên tên biến ở đâycó thể là bất cứ thứ gì, không cần phải là khóa hoặc ngôn ngữ, nó có thể là bất cứ thứ gì bạn muốn và điều này quen thuộc với bạn vì đó là cách chúng tôi xác định các cặp giá trị khóa trong mảng khi bạn tạo mảng

A computer screen shot of a program

Description automatically generated

Trong trường hợp này thì đây là một mảng được lập chỉ mục nên khóa sẽ chỉ là các chỉ mục số nên đây sẽ là 0 1 2 3 và 4. vì vậy chúng ta có thể thực hiện khóa và làm mới, chúng ta nhận được 0 1 2 3 và 4.

A screenshot of a computer

Description automatically generated

Bây giờ như tôi đã đề cập trong mỗi lần lặp,phần tử hiện tại của mảng được gán cho biến này ở đây và nó được gán bởi một giá trị, tuy nhiên bạn có thể gán giá trị đó bằng tham chiếu bằng cách thêm ký hiệu và ở đây A screen shot of a computer program

Description automatically generated



Và về cơ bản điều này chỉ gán giá trị của hiện tại phần tử theo tham chiếu, điều đó có nghĩa là nếu bạn sửa đổi phần này, nó sẽ thực sự sửa đổi mảng ban đầu. Chẳng hạn như chúng ta có thể đặt ngôn ngữ thành php và sau đó chúng ta có thể in ngôn ngữ lập trình của mình ngay sau đó và chúng ta thấy rằng tất cả các phần tử đều là php ngay bây giờ

A screenshot of a computer

Description automatically generated



Nếu chúng ta không thực hiện bằng tham chiếu mà thực hiện theo giá trị khi làm mới trang thì mảng ban đầu không bị thay đổi. Lưu ý nhanh ở đây là biến mảng ở đây trong trường hợp này biến ngôn ngữ sẽ vẫn giữ nguyên sau khi vòng lặp foreach đã hoàn tất.

A screenshot of a computer

Description automatically generated



Hãy để tôi xóa “print\_r($programingLanguages);” và chúng ta có thể thực hiện ngôn ngữ echo ngay sau vòng lặp và hãy giữ câu lệnh echo này ở đây nếu chúng ta làm mới trang:

A screenshot of a computer

Description automatically generated



chúng ta nhận được 0 1 2 3 4 và sau đó chúng ta cũng bị biến “rust”, biến này không bị phá hủy sau khi chạy vòng lặp foreach. Có nghĩa là nếu bạn quyết định sử dụng biến “$language” ở một nơi khác hoặc bạn mắc lỗi đánh máy và bạn sử dụng biến này thay vì một biến khác thì điều này có thể gây ra một số vấn đề trong mã của bạn,đặc biệt là khi bạn làm việc với các tham chiếu.

A computer screen shot of a program

Description automatically generated



Vì vậy khi bạn đang làm việc với thứ gì đó như thế này, biến này sẽ không bị hủy. Do đó nếu bạn làm điều gì đó như “$language = ‘php’;” ở đây và sau đó chúng tôi in ngôn ngữ lập trình của mình “print\_r($programmingLanguages);”, bạn sẽ thấy mục cuối cùng được đổi thành PHP vì ngôn ngữ này vẫn là tham chiếu đến phần tử cuối cùng trong mảng và trong trường hợp này, nó là rush và rush được đổi thành PHP

A screenshot of a computer

Description automatically generated

Điều tôi làm trong những trường hợp này là nếu tôi quyết định sử dụng tham chiếu ở đây sau đó mỗi vòng lặp tôi thường chỉ hủy biến bằng cách sử dụng “unset($language);” và theo cách đó tôi chắc chắn rằng điều này sẽ không ghi đè lên bất kỳ thứ gì trong ngôn ngữ lập trình nữa vì vậy nếu tôi làm mớibây giờ, chúng tôi sẽ nhận được rush trở lại A screenshot of a computer screen

Description automatically generated

Một trường hợp sử dụng tốt khác cho vòng lặp foreach là lặp lại trên các mảng kết hợp vì vậy, ví dụ: giả sử chúng tôi có một biến “$user” và giả sử chúng tôi có một số thuộc tính ở đây như tên email và kỹ năng, sau đó chúng tôi có thể sử dụng vòng lặp foreach để lặp lại các thuộc tính đó để chúng tôi có thể thực hiện cho từng người dùng làm giá trị khóa và sau đó chúng tôi có thể lặp lại khóa tất cả về giá trị và một dòng mới: A screenshot of a computer

Description automatically generated

Tôi đang nhận được tên và email nhưng chúng tôi đang nhận được mảng cảnh báo này để chuyển đổi chuỗi và và nhận được giá trị là một mảng như mong đợi. Để giải quyết vấn đề này bằng cách sử dụng thứ gọi là json\_encode để bạn có thể mã hóa nó thành json và in nó ra

A screen shot of a computer

Description automatically generated



Theo cách khác mà bạn có thể sử dụng implode nhưng tôi sẽ chỉ sử dụng implode nếu đó là một mảng vì vậy trước tiên tôi sẽ kiểm tra xem có phải là mảng không giá trị giá trị bằng với giá trị implode và sau đó chỉ cần in nó ra theo cách nàyA screen shot of a computer

Description automatically generated

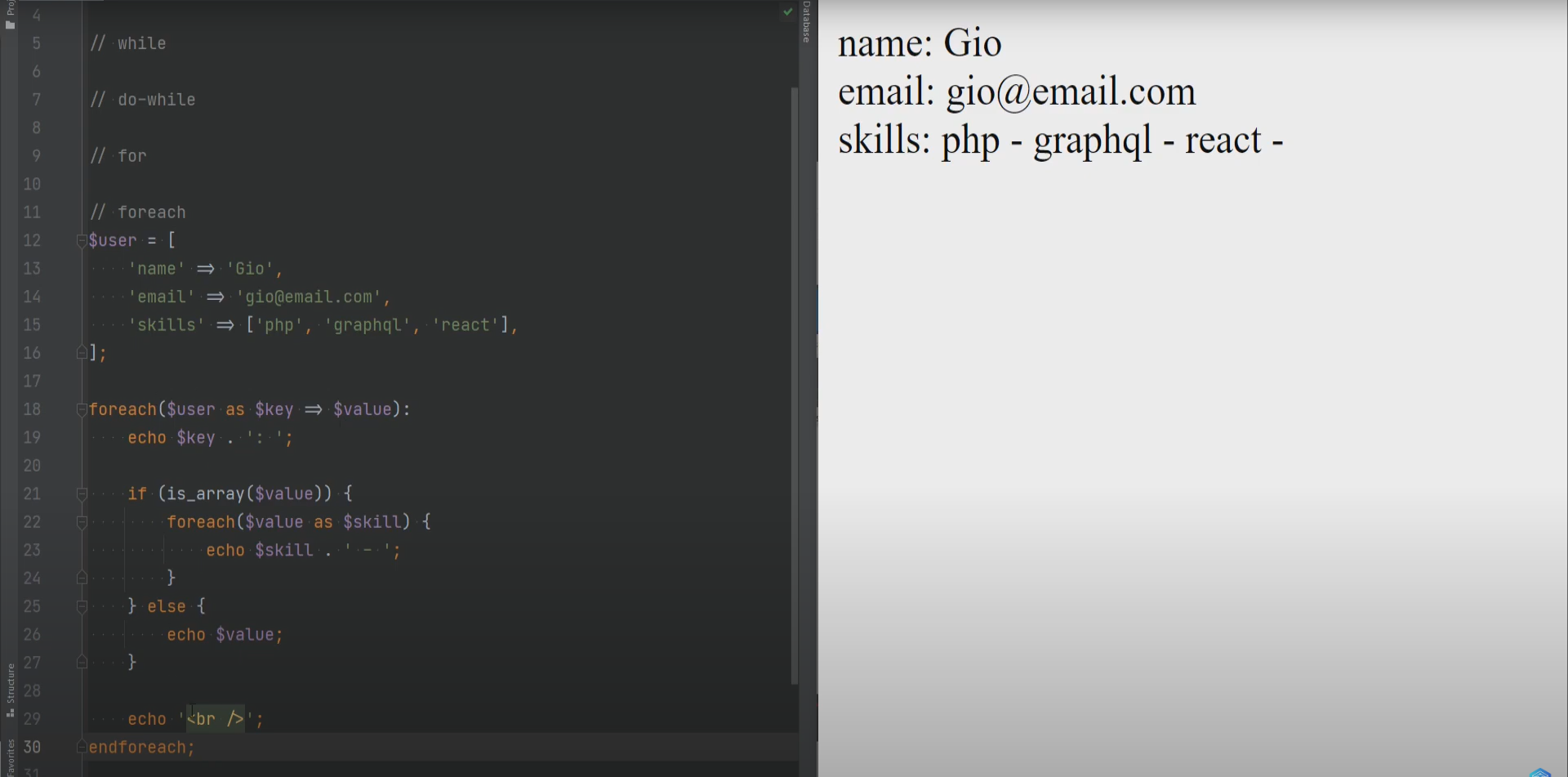


Hoặc bạn có thể chỉ cần lặp lại giá trị mà bạn có thể lồng vào mỗi vòng lặp ở đây cho mỗi giá trị “$skill” và chỉ cần lặp lại “$skill” và một loại dấu phân cách nào đó rồi xóa “if (is\_array($value))” thay bằng “echo $key . ‘: ‘;” và xóa  khỏi đây. Nếu nó không phải là một mảng, chúng tôi chỉ có thể lặp lại giá trị như hiện tại nếu bạn làm mới trang

A screenshot of a computer

Description automatically generated

Mọi thứ hoạt động như mong đợi nhưng điều đó thực sự tùy thuộc vào cách bạn muốn xử lý những điều này. Trong các trường hợp, cũng có một cú pháp thay thế cho vòng lặp foreach và for vì vậy thay vì dấu ngoặc nhọn, bạn có thể thực hiện dấu hai chấm và thay vì dấu ngoặc nhọn cuối bạn có thể dùng “endforeach” và chúng tôi làm mới nó hoạt động theo cách tương tự.



Và vòng lặp for cũng tương tự như vậy nên ở đây bạn cũng có thể có cú pháp thay thế bằng cách sử dụng dấu hai chấm và endfor ở cuối và nó sẽ hoạt động theo cách tương tự

A screenshot of a computer

Description automatically generated

Vì vậy, cảm ơn bạn đã xem, hãy nhấn like và đăng ký. Và tôi sẽ gặp bạn trong video tiếp theo.