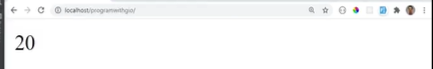
**PHP Operator Precedence & Associativity - Full PHP 8 Tutorial**Bạn có thể kết hợp các toán tử để tạo thành các biểu thức phức tạp. Khi sử dụng nhiều toán tử  
trong một biểu thức, chúng được nhóm theo tiền lệ của chúng. Và nếu họ có cùng tiền lệ, sau đó tính liên kết của chúng quyết định cách chúng được nhóm lại.

  
Bảng này được sắp xếp theo các tiền lệ, vì vậy những bảng hiển thị ở trên có giá trị cao hơn, tiền lệ hơn những gì được hiển thị ở phía dưới. Và ở phía bên trái ở đây là sự kết hợp và chúng ta sẽ nói về điều này sau một phút.  
Hãy lấy ví dụ đơn giản nhất.

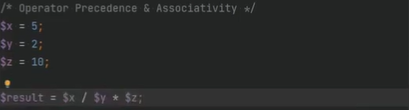
C:\Users\ADMIN\OneDrive\Hình ảnh\Ảnh chụp màn hình\Ảnh chụp màn hình 2023-09-08 113304.png  
Nếu chúng ta có X bằng năm cộng 3 nhân 5, nếu không có quyền ưu tiên, thì cái này sẽ thực hiện như 5 cộng 3, bằng 8, rồi nhân 5, 40. Nhưng vì chúng ta có quyền ưu tiên nên 3 nhân 5 được thực hiện trước vì phép nhân có độ ưu tiên cao hơn phép trừ, và do đó điều này được nhóm lại với nhau và được thực hiện trước tiên. Ba lần 5 trả về 15, rồi chúng ta thực hiện phép cộng, bằng 20.  
Như bạn nhận thấy, bài tập có mức thấp nhất được ưu tiên hơn tất cả những điều này, ngay tại bảng trên. Vì vậy, đó là lý do tại sao mọi thứ ở đây đều có được thực hiện trước và chỉ sau đó việc chuyển nhượng mới được thực hiện. Câu trả lời cuối cùng sẽ là 20  
Nếu chúng tôi sử dụng EchoX, chúng tôi sẽ nhận được 20.



  
Bạn cũng có thể buộc các tiền lệ bằng cách sử dụng dấu ngoặc đơn. Nếu bạn muốn năm cộng một được thực hiện trước, bạn có thể đặt nó trong ngoặc đơn nhưcái này và bây giờ cái này sẽ được thực thi trước.  
  
Câu trả lời cuối cùng sẽ là 40.

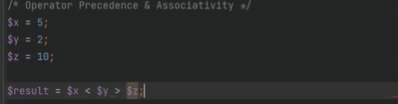
C:\Users\ADMIN\OneDrive\Hình ảnh\Ảnh chụp màn hình\Ảnh chụp màn hình 2023-09-08 114143.png  
Bây giờ, tôi đã đề cập đến tính kết hợp trước đó, và điều đó về cơ bản có nghĩa là nếu các toán tử có cùng tiền lệ, sau đó tính liên kết của chúng quyết định cách chúng được nhóm lại. Như bạn có thể thấy trong bảng trên, chúng ta có thể có các toán tử kết hợp bên phải, chúng ta có thể có các toán tử kết hợp bên trái, và chúng ta có các toán tử không có tính kết hợp. Hãy để tôi chỉ cho bạn một ví dụ về ý nghĩa của sự kết hợp.

C:\Users\ADMIN\OneDrive\Hình ảnh\Ảnh chụp màn hình\Ảnh chụp màn hình 2023-09-08 114657.png  
Hãy làm X bằng Y bằng năm. Ở đây, vì độ ưu tiên của toán tử giống nhau vì chúng ta đang sử dụng cùng một toán tử, thì tính kết hợp sẽ xác định cách chúng được nhóm lại. Vì bài tập có quyền tính kết hợp, điều đó có nghĩa là nó được thực thi từ phải sang trái. Do đó, Y bằng năm được nhóm lại cùng nhau, và sau đó X bằng bất cứ giá trị nào mà biểu thức này ước tính.  
Hãy làm một ví dụ khác.



Giả sử X bằng 5, Y bằng 2 và Z bằng 10. Khi đó chúng ta có thể có kết quả bằng X chia cho Y nhân Z. Nếu chúng ta nhìn vào bảng ở đây, chúng ta thấy rằng phép nhân và phép chia có cùng tiền lệ, và do đó tính kết hợp quyết định cách chúng được nhóm lại. Tính kết hợp của các toán tử này còn lại. Điều đó có nghĩa là nó đi từ trái sang phải. Do đó, X chia cho Y được nhóm lại đầu tiên, sau đó kết quả của biểu thức đó được nhân với Z. Và cuối cùng, kết quả cuối cùng được gán cho biến result.

Các toán tử có cùng mức độ ưu tiên và không liên kết có thể được sử dụng cạnh nhau. Ví dụ: bạn không thể có thứ gì đó như X nhỏ hơn Y lớn hơn Z.

  
Điều này sẽ không hoạt động. Tuy nhiên, bạn có thể sử dụng thứ gì đó có mức độ ưu tiên khác.  
Ví dụ, các toán tử so sánh cócác tiền lệ khác nhau và sau đó bạn có thể sử dụng tiền lệ đó trong cùng một biểu thức.

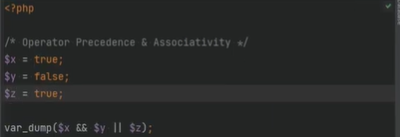
  
Bạn có thể làm điều này. Hãy làm một số ví dụ thực tế với các toán tử logic. Ví dụ: nếu chúng ta có X bằng true và Y bằng false, và sau đó chúng ta đã đổ cột X chứ không phải Y



chúng ta sẽ đạt được kết quả đúng.



Hãy xem các toán tử này thực sự được nhóm như thế nào. Nếu chúng ta xem xét bảng ở trên, chúng ta sẽ thấy rằng toán tử một ngôi có giá trị cao hơn được ưu tiên hơn các toán tử logic khác và do đó nó được nhóm lại đầu tiên và nó được thực thi trước, sau đó kết quả của nó sẽ được so sánh với X.  
Một ví dụ khác là hãy thực hiện X, N, Y hoặc Z và gán Z thành true.



Ở đây toán tử kết thúc có độ ưu tiên cao hơn toán tử hoặc. Biểu thức này được nhóm lại và điều này sẽ được thực thi trước và sau đó kết quả của việc đó sẽ được xử lý bằng biến Z. Kết quả của điều đó là sai và sau đó nó sẽ sai hoặc đúng và kết quả cuối cùng sẽ đúng. Nếu chúng tôi làm mới trang, chúng tôi nhận được sự thật.

Như đã đề cập trong video trước, chúng ta có hai biến thể của toán tử logic. Chúng ta có phần cuối này

C:\Users\ADMIN\OneDrive\Hình ảnh\Ảnh chụp màn hình\Ảnh chụp màn hình 2023-09-08 120458.png

và một phiên bản khác là phần cuối này ngay tại đây.

C:\Users\ADMIN\OneDrive\Hình ảnh\Ảnh chụp màn hình\Ảnh chụp màn hình 2023-09-08 120515.png

Chúng ta cũng có cái này

C:\Users\ADMIN\OneDrive\Hình ảnh\Ảnh chụp màn hình\Ảnh chụp màn hình 2023-09-08 120910.png

hoặc và chúng ta có cái này hoặc ngay đây.

C:\Users\ADMIN\OneDrive\Hình ảnh\Ảnh chụp màn hình\Ảnh chụp màn hình 2023-09-08 120949.png

Như tôi đã đề cập trong video trước, sự khác biệt là ở chỗ, như bạn có thể thấy, những cái có tiền lệ thấp nhất

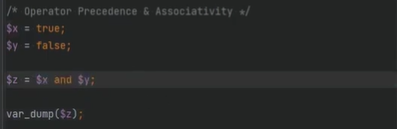


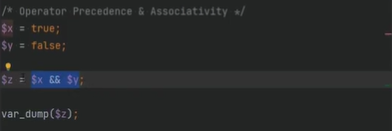
và những tiền lệ này có tiền lệ cao hơn.



Để cho bạn thấy điều này trong thực tế,  
hãy làm X bằng true và sau đó Y bằng false và sau đó Z bằng X và Y và hãy dùng VARDOMP Z và chúng ta sẽ nhận được sai vì điều này hoạt động như mong đợi. Để cho bạn thấy điều này trong thực tế, hãy làm X bằng true và sau đó Y bằng false và sau đó Z bằng X và Y và hãy dùng VARDOMP Z

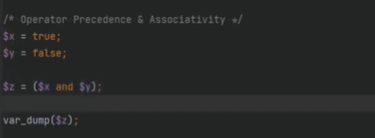


và chúng ta sẽ nhận được sai vì điều này hoạt động như mong đợi.  
C:\Users\ADMIN\OneDrive\Hình ảnh\Ảnh chụp màn hình\Ảnh chụp màn hình 2023-09-08 121622.png  
Tuy nhiên, nếu chúng ta thay đổi điều này thành end, như bạn thấy, toán tử gán  
  
  
ở đây có mức độ ưu tiên cao hơn phần cuối và mức độ ưu tiên thấp hơn phần cuôi. Trong ví dụ trước



khi chúng ta đã kết thúc, việc này được nhóm lại với nhau và được thực thi trước rồi mới thực hiện nhiệm vụ. Nhưng trong trường hợp từ khóa end, từ khóa end có độ ưu tiên thấp hơn toán tử gán và do đó, nó được nhóm lại với nhau và Z bằng nhau đến X, điều này đúng trong trường hợp này và về cơ bản điều này sẽ bị loại bỏ. Z bây giờ sẽ bằng true và nếu chúng ta làm mới trang, chúng ta sẽ nhận được giá trị true..

Để tránh nhầm lẫn và kết quả không mong muốn, rất khuyến khích sử dụng dấu ngoặc đơn vì hai lý do. Một là nó sẽ làm rõ hơn và khả năng đọc mã của bạn và bạn cũng đang nói rõ về cách bạn muốn các toán tử được nhóm lại thay vì dựa vào tiền lệ và tính kết hợp.

Trong trường hợp này, nếu chúng ta đặt cái này trong ngoặc đơn như thế này thì chúng ta sẽ không gặp phải vấn đề đó.  
  
Nó sẽ trả về sai. Điều này kết thúc video của chúng tôi về các nhà khai thác. Còn một số toán tử nữa mà tôi chưa đề cập đến trong các video này vì chúng ta chưa đi đến phần đó của khóa học. Trong các video tiếp theo, chúng ta sẽ chuyển sang cấu trúc điều khiển. Cảm ơn bạn rất nhiều vì đã xem.