# Bash conditional expressions

Lệnh test trong cách viết kịch bản của Linux sẽ luôn thực hiện công việc so sánh giữa các giá trị và luôn trả về kết quả là True hay False. Kết quả True hay False có được chính là điều kiện mỗi khi các Bạn thực hiện công việc so sánh.

Thông thường khi kết hợp lệnh test cùng với mệnh đề if để đơn giản hóa cú pháp khá phiền phức bởi cặp dấu ngoặc vuông của mệnh đề if...else, cách sử dụng cú pháp của lệnh test.

**test**<câu lệnh> hoặc <biểu thức>

Câu lệnh kết hợp với mệnh đề if như sau:

Mệnh đề if chuẩn:

if [ điều kiện ]

then

Các câu lệnh của if

fi

Câu lệnh test hoặc cặp dấu [ expr ]

Mệnh đề if kết hợp với câu lệnh test

if test <điều kiện>

then

Các câu lệnh của if

fi

Cả 2 Câu lệnh thường đc sử dụng để kiểm tra định dạng file hoặc so sánh các giá trị. Thường thì đc sử dụng cho việc:

-So sánh các thuộc tính của file

-Thực hiện việc so sánh các chuỗi

-So sánh toán học

Cú pháp:

test condition

condition là một loạt các tham số và giá trị đc test thực hiện. Nếu sử dụng với if-then thì sẽ tương tự như vầy:

if test condition

then

commands

fi

hoặc

if [ condition ]

then

commands

fi

\*\*\*Lưu ý khoảng trắng của condition

Một số phép so sánh trong shell:

|  |  |
| --- | --- |
| -eq | ngang bằng |
| -ne | không ngang bằng |
| -lt | nhỏ hơn |
| -le | nhỏ hơn hoặc bằng |
| -gt | lớn hơn |
| -ge | lớn hơn hoặc bằng |

Một số phép xử lý thuộc tính file:

|  |  |
| --- | --- |
| -s | file tồn tại và khác rỗng(NULL) |
| -f | file tồn tại và ko phải là thư mục |
| -d | thư mục tồn tại |
| -x | file có thể thực thi |
| -w | file có thể ghi |
| -r | file có thể đọc |

Làm việc với string:

|  |  |
| --- | --- |
| Operator | Meaning |
| string1 = string2 | string1 ngang bằng string2 |
| string1 != string2 | string1 khác string2 |
| string1 | string1 ko rỗng hoặc ko đc định nghĩa |
| -n string1 | string1 tồn tại và không rỗng |
| -z string1 | string1 tồn tại nhưng rỗng |

Phép toán logic:

|  |  |
| --- | --- |
| Operator | Meaning |
| ! expression | phép NOT |
| expression1 -a expression2 | phép AND |
| expression1 -o expression2 | phép OR |

# Directory stacks

Nếu bạn sử dụng csh, bạn sẽ biết về directory stacks. Về cơ bản, Directory stacks là 1 stack để lưu trữ các thư mục. Để tương tác với Directory stacks, bạn sử dụng lệnh pushd để thêm 1 thư mục vào trong stack và chuyển tới thư mục mới, lệnh popd lấy 1 thư mục ra khỏi stack và chuyển tới thư mục đó. Lệnh dirs hiển thị tất cả các thư mục có trong directory stacks.

# Kiểm tra bash shell script có đang chạy dưới quyền root hay không

Khi một tài khoản user được tạo ra, một user ID cũng sẽ được cấp cho mỗi tài khoản đó. BASH shell ghi lại user ID này vào biến $UID. Những gì thuộc về user ID này sẽ được lưu trong biến $EUID. Vì thế ta có thể kiểm tra bằng cách:

#!/bin/bash

# Init

FILE="/tmp/out.$$"

GREP="/bin/grep"

#....

# Make sure only root can run our script

if [[ $EUID -ne 0 ]]; then

echo "This script must be run as root" 1>&2

exit 1

fi

# ...

Mount /dev/sdb1 only if you are a root

#!/bin/bash

if [[ $EUID -ne 0 ]]; then

echo "You must be a root user" 2>&1

exit 1

else

mount /dev/sdb1 /mnt/disk2

fi

Posted by: Vivek Gite

# stderr và stdout

Trong Unix shell, nếu ta muốn kết hợp stderr và stdout , ta có thể thêm vào cuối command cụm: "2>&1". Đây là cấu trúc POSIX shell.

- 2: "Standard error", mô tả tập tin đầu ra.

- >&: Nhân đôi một đối tượng gọi là Output File Descriptor operator (một biến thể của Output Redirection operator >). Nếu [x]>&[y] tức là tập tin được kí hiệu bởi x là một bản copy từ y.

- 1: "Standard output": output file descirptor (tập mô tả đầu ra).

- biểu thức 2>&1 copy file descriptir đến vị trí 2, do đó bất kỳ output nào được viết bởi 2 (standard error - stderr) cũng sẽ được viết vào 1 (standard output – stdout).

0 - stdin

1 - stdout

2 - stderr

& - để biểu thị đây là một file descriptor, không phải một tên fime bình thường.

# getopts

- đưa ra các tuỳ chọn phân tích cú pháp từ dòng lệnh của shell script.

- getopts có được các tuỳ chọn và các đối số đi kèm với chúng từ một danh sách các tham số tuân theo chuẩn POSIX.2 (các chữ cái đơn đứng trước bởi dấu - và có thể theo sau bởi một đối số, các chữ cái đơn có thể được nhóm lại).

- 'optstring' đưa ra tất cả các chữ cái của tuỳ chọn. Ví dụ nếu ta muốn có các tuỳ chọn là -a, -f và -s thì optstring là afs. Nếu muốn một chữ cái có kèm theo đối số phía sau hoặc nhóm các chữ cái thì thêm dấu : vào sau chữ cái đó, ví dụ như a:fs (nghĩa là getopts mong muốn tuỳ chọn -a có dạng '-a value').

- Thường thì có khoảng trắng để tách biệt giữa chữ cái đại diện cho tuỳ chọn và đối số của nó nhưng getopts cũng xử lý các các tuỳ chọn mà chữ cái gắn liền với đối số, ví dụ như: '-avalue'.

- optstring không thể chứa ký tự ?

- Tên trên dòng lệnh command của getopts là tên của một biến shell. Mỗi lần gọi getopts, nó sẽ có được các tuỳ chọn tiếp theo từ các tham số vị trí và đặt ký tự tuỳ chọn vào tên biến shell.

- getopts đặt một dấu ? trong tên nếu nó tìm thấy một tuỳ chọn không được khai báo trong optstring, hoặc nếu giá trị tùy chọn đó bị thiếu.

- Mỗi tuỳ chọn trên dòng lệnh command có một chỉ mục. Tuỳ chọn đầu tiên có chỉ mục là 1, tuỳ chọn thứ hai có chỉ mục là 2 v.v. Khi getopts nhận được một tùy chọn từ dòng lệnh, nó sẽ lưu các chỉ mục của đối số tiếp theo để xử lý trong các biến shell OPTIND.

- Khi một ký tự tuỳ chọn có một đối số đi kèm (được chỉ định với a: trong optstring), getopts lưu các đối số như một chuỗi string trong biến shell OPTARG. Nếu một option không có đối số, hoặc getopts yêu cầu một đối số nhưng không tìm thấy thì getopts sẽ không đặt (unset) OPTARG.

- Khi getopts đạt đến cuối các tuỳ chọn và exit với status value = 1. Nó đặt tên cho các đặc tính ? và đặt (set) OPTIND vào chỉ mục của đối số đầu tiên sau các tuỳ chọn.

- getopts công nhận sự kết thúc của các tùy chọn khi thoả mãn một trong các điều kiện sau:

- một đối số không bắt đầu bằng dấu -

- một đối số đặc biệt được ký hiệu là --, đây là đối số đánh dấu sự kết thúc của các tuỳ chọn.

- error (ví dụ như một ký tự tuỳ chọn không được công nhận).

- OPTIND và OPTARG là các cục bộ của shell script (cần được khai báo rõ ràng với giá trị khởi tạo).

- Theo mặc định, getopts sẽ xuất ra thông báo lỗi khi nó tìm thấy một tuỳ chọn không được công nhận hoặc một lỗi nào đó. Nếu không muốn thấy thông báo này, chỉ cần thêm ký tự : vào đầu của ký tự đầu tiên trong optstring.

- ngoài tiêu chuẩn POSIX.2, getopts cũng có thể nhận được các chữ cái tuỳ chọn mà có dấu + ở đầu.

- Nếu optstring bắt đầu bằng dấu :+ hoặc +, hoặc tuỳ chọn có dấu + ở đầu, biến OPTARG vẫn được đặt như bình thường.

- Nếu optstring bắt đầu bằng dấu :++ hoặc ++, hoặc dấu + xuất hiện ở bất cứ vị trí nào trong optstring ... thì không thể có các option với dấu + ở đầu và có đối số.

- Các biến môi trường:

- OPTARG: lưu các giá trị của đối số của tùy chọn được tìm thấy trong getopts.

- OPTIND: chứa các chỉ mục của đối số tiếp theo sẽ được xử lý.

- Các exit status value (giá trị trạng thái thoát) có thể có:

- 0: getopts tìm thấy một đối số script command line có dạng của một tuỳ chọn. (xảy ra cho dù nó có công nhận tuỳ chọn đó hay không).

- 1: getopts đi đến cuối của các tùy chọn, hoặc xảy ra lỗi.

- 2: Thất bại vì dòng lệnh tuỳ chọn có dạng không hợp lệ.

# Kiểm tra xem package đã được cài đặt chưa

systemctl status package\_name