Bash scripts

Привет, мир!

По традиции напишем первый сценарий, выводящий всем известную фразу: "Привет, мир!" (Погладь кота). Для редактирования сценариев вы можете использовать любой текстовый редактор.

#!/bin/bash

echo "Кису любишь!"

Первая строка нашего сценария — это указание, что он должен быть обработан программой /bin/bash. Обратите внимание: если между # и ! окажется пробел, то данная директива не сработает, поскольку будет воспринята как обычный комментарий.

# Комментарий

Вторая строка — это оператор echo, выводящий нашу строку. Сохраните сценарий под именем kisa и введите команду:

$ chmod +х kisa

Для запуска сценария введите команду:

./kisa

На экране вы увидите строку:

Кису любишь!

Чтобы вводить для запуска сценария просто kisa (без ./), сценарий нужно скопировать в каталог /usr/bin (точнее, в любой каталог из переменной окружения ратн):

# cp ./kisa /usr/bin

Использование переменных

В любом серьезном сценарии вы не обойдетесь без использования переменных. Переменные можно объявлять в любом месте сценария, но до места их первого применения. Рекомендуется объявлять переменные в самом начале сценария, чтобы потом не искать, где вы объявили ту или иную переменную.

Для объявления переменной используется следующая конструкция:

переменная=значение

Пример объявления переменной и использования:

KISA=Киса

echo $KISA

Обратите внимание на следующие моменты:

При объявлении переменной знак доллара не ставится, но он обязателен при использовании переменной;

При объявлении переменной не должно быть пробелов до и после знака =.

Значение для переменной указывать вручную не обязательно — его можно прочитать с клавиатуры:

read KISA

или со стандартного вывода программы:

KISA='hostname'.

В скриптах часто используются переменные окружения. Вот примеры некоторых часто используемых переменных окружения:

BASH - полный путь до исполняемого файла командной оболочки BASH

BASH\_VERSION

HOME - домашний каталог пользователя, который запустил сценарий;

HOSTNАМЕ - имя компьютера;

RANDOM - случайное число в диапазоне от 0 до 32 767

OSTYPE - тип операционной системы

PWD - текущий каталог

PS1 - строка приглашения

UID - ID пользователя, который запустил сценарий

USER - имя пользователя.

Для установки собственной переменной окружения используется команда export:

# после этого переменная KISA будет доступна в других сценариях export $KISA

Передача параметров сценарию

Для передачи параметров используются следующие специальные переменные:

$0 - содержит имя сценария

$n - содержит значение параметра (n — номер параметра)

$# - позволяет узнать количество параметров, которые были переданы.

Рассмотрим небольшой пример обработки параметров сценария.

# Сценарий должен вызываться так:

# имя\_сценария параметр

# Анализируем первый параметр

case "$1" in

start)

# Действия при получении параметра start

echo "START"

stop)

# Действия при получении параметра stop

echo "STOP"

;;

# Действия в остальных случаях

\*)

# Выводим подсказку о том, как нужно использовать сценарий, и

# завершаем работу сценария

echo "Интструкция: $0 {start|stop}"

exit: 1

;;

esac

Массивы

Массивы объявляются подобно переменным.

A[0]=1

А[1]=2

echo $A[0]

Циклы

Как и в любом языке программирования, в Bash можно использовать циклы. Мы рассмотрим циклы for и while, хотя вообще в Bash доступны также циклы until и select, но они применяются довольно редко.

Синтаксис цикла For выглядит так:

for переменная in список

do

команды

done

В цикле при каждой итерации переменной будет присвоен очередной элемент списка, над которым будут выполнены указанные команды. Чтобы было понятнее, рассмотрим примеры:

for i in 1 2 3 4 5

do

echo "Pardon $i times"

done

#!/bin/bash

for i in {1..5}

do

echo "Sorry $i times"

done

for i in {0..10..2}

do

echo "Welcome $i times"

done

Синтаксис цикла while выглядит похоже:

while условие

do

команды

done

Цикл while выполняется до тех пор, пока истинно заданное условие.

#!/bin/bash

x=1

while [ $x -le 5 ]

do

echo "Welcome $x times"

x=$(( $x + 1 ))

done

Условные операторы

В Bash доступно два условных оператора, if и case. Синтаксис оператора if следующий:

if [ conditional expression ]

then

statements

fi

if [ conditional expression ]

then

statements

else

statements

fi

if [ conditional expression1 ]

then

statements

elif [ conditional expression2 ]

then

statements

else

statements

fi

Ответственная задача — это правильно составить условие. Условия записывается в квадратных скобках.

[N == 10]

[N != 10]

Операции сравнения указываются не с помощью привычных знаков > или <,

а с помощью следующих выражений:

-lt - меньше;

-gt - больше;

-1е - меньше или равно;

-ge - больше или равно;

-eq - равно (используется вместо ==).

-e file - условие истинно, если файл существует;

-d directory - условие истинно, если каталог существует;

-x - условие истинно, если файл является исполнимым.

Оператор case:

case expression in

pattern1 )

statements ;;

...

patternN )

statements ;;

…

\*)

statements ;;

…

esac

Значение указанного выражения по очереди сравнивается с приведенными значениями ( pattern1 ... patternN). Если есть совпадение, то будут выполнены команды, соответствующие значению. Если совпадений нет, то будут выполнены команды по умолчанию.

Функции

В Bash можно использовать функции. Синтаксис объявления функции следую­ щий:

имя ( ) { список; }

Вот пример объявления и использования функции:

list\_txt()

{

echo "Выводим текстовые файлы"

ls \*.txt

}

Сценарий мониторинга журнала

Системные журналы постоянно обновляются, а наш сценарий будет каждые 3 секунды выводить последние 15 строк выбранного вами журнала. Сценарий будет полезен при настройке системы, когда нужно постоянно просматривать журналы, чтобы понять, как система реагирует на новые настройки. Для прекращения работы сценария нужно нажать клавиши <Ctrl>+<С>.

#!/bin/bash

# Интервал обновления, в секундах

INT=3

while [true]

do

# Выводим последние 15 строк журнала

tail -n 15 $1

# Ждем

sleep $INT

# Две пустые строки

echo; echo

done

Формат вызова следующий:

./сценарий файл\_журнала

Переименование файлов

Следующий сценарий сложнее: он ищет файлы в текущем каталоге, в именах которых есть пробелы, и заменяет пробелы на символы подчеркивания.

#!/bin/bash

#

# Сколько файлов мы переименовали

num=0

# Перебираем все файлы в текущем каталоге

for filename in \*

do

# Передаем имя файла фильтру grep

# Если имя файла содержит пробел, то код завершения

# последней операции равен 0

echo "$filename" | grep -q " "

if [ $? -eq 0 ]

then

# -Если код завершения равен 0, переименовываем файл

fname=$filename

n='echo $fname | sed -e "s/ /\_/g"'

mv "$fname" $n

let "num += 1"

fi

done

echo "Переименовано файлов: $num"

exit 0

Преобразование систем счисления

Сейчас мы напишем простенький сценарий, преобразующий десятичное число в шестнадцатеричное с помощью программы dс . Самим преобразо­ ванием будет заниматься программа dс, а наш сценарий только подготовит данные для этой программы.

#!/bin/bash

B=16 # Основание системы счисления

# Проверяем, является ли параметр $1 числом

if [ -z "$1" ]

then

echo "Использование: команда число"

exit 1

fi

# Передаем число ($1) , систему счисления

# программе dc

echo '"$1" "$В" o p'|dc