Table Of Contents

nixCraft: Linux Tips, Hacks, Tutorials, And Ideas In Blog Format
http://www.cyberciti.biz/

Bash For Loop Examples

Posted by Vivek Gite <vivek@nixcraft.com>

[1]

How do I use bash for loop to repeat certain task under Linux / UNIX operating system? How do I set infinite loops using for statement? How do I use three-parameter for loop control expression?

A 'for loop' is a bash programming language statement which allows code to be repeatedly executed. A for loop is classified as an iteration statement i.e. it is the repetition of a process within a bash script.

For example, you can run UNIX command or task 5 times or read and process list of files using a for loop. A for loop can be used at a shell prompt or within a shell script itself.

[2]

for loop syntax

Numeric ranges for syntax is as follows:

```
for VARIABLE in 1 2 3 4 5 .. N
do
 command1
 command2
 commandN
done
```

This type of for loop is characterized by counting. The range is specified by a beginning (#1) and ending number (#5). The for loop executes a sequence of commands for each member in a list of items. A representative example in BASH is as follows to display welcome message 5 times with for loop:

```
#!/bin/bash
for i in 1 2 3 4 5
do
   echo "Welcome $i times"
done
```

Sometimes you may need to set a step value (allowing one to count by two's or to count backwards for instance). It can be done easily with seq command [3]. A representative example in bash as follows:

```
#!/bin/bash
for i in $(seq 1 2 20)
do
   echo "Welcome $i times"
done
```

Latest bash version 3.0+ has inbuilt support for setting up ranges:

```
#!/bin/bash
for i in {1..5}
do
   echo "Welcome $i times"
done
```

Bash v4.0+ has inbuilt support for setting up a step value using {START..END..INCREMENT} syntax:

```
#!/bin/bash
echo "Bash version ${BASH_VERSION}..."
```

```
for i in {0..10..2}
  do
     echo "Welcome $i times"
 done
```

Sample outputs:

```
Bash version 4.0.33(0)-release...
Welcome 0 times
Welcome 2 times
Welcome 4 times
Welcome 6 times
Welcome 8 times
Welcome 10 times
```

4.0.33(0)-release

Three-expression bash for loops syntax

This type of for loop share a common heritage with the C programming language. It is characterized by a three-parameter loop control expression; consisting of an initializer (EXP1), a loop-test or condition (EXP2), and a counting expression (EXP3).

```
for (( EXP1; EXP2; EXP3 ))
do
 command1
 command2
 command3
done
```

A representative three-expression example in bash as follows:

```
#!/bin/bash
for (( c=1; c<=5; c++ ))
do
 echo "Welcome $c times..."
done
```

Sample output:

```
Welcome 1 times
Welcome 2 times
Welcome 3 times
Welcome 4 times
Welcome 5 times
```

How do I use for as infinite loops?

Infinite for loop can be created with empty expressions, such as:

```
#!/bin/bash
for (( ; ; ))
do
   echo "infinite loops [ hit CTRL+C to stop]"
done
```

Conditional exit with break

You can do early exit with break statement inside the for loop. You can exit from within a FOR, WHILE or UNTIL loop using break. General break statement inside the for loop:

```
for I in 1 2 3 4 5
done
  statements1      #Executed for all values of ''I'', up to a disaster-condition if any.
```

```
  statements2
  if (disaster-condition)
  then
 break           #Abandon the loop.
  fi
  statements3          #While good and, no disaster-condition.
done
```

Following shell script will go though all files stored in /etc directory. The for loop will be abandon when /etc/resolv.conf file found.

```
#!/bin/bash
for file in /etc/*
do
 if [ "${file}" == "/etc/resolv.conf" ]
 then
  countNameservers=$(grep -c nameserver /etc/resolv.conf)
  echo "Total  ${countNameservers} nameservers defined in ${file}"
  break
 fi
done
```

Early continuation with continue statement

To resume the next iteration of the enclosing FOR, WHILE or UNTIL loop use continue statement.

```
for I in 1 2 3 4 5
done
  statements1      #Executed for all values of ''I'', up to a disaster-condition if any.
  statements2
  if (condition)
  then
 continue   #Go to next iteration of I in the loop and skip statements3
  fi
  statements3
done
```

This script make backup of all file names specified on command line. If .bak file exists, it will skip the cp command.

```
#!/bin/bash
FILES="$@"
for f in $FILES
do
        # if .bak backup file exists, read next file
 if [ -f ${f}.bak ]
 then
  echo "Skiping $f file..."
  continue  # read next file and skip cp command
 fi
        # we are hear means no backup file exists, just use cp command to copy file
 /bin/cp $f $f.bak
done
```

Recommended readings:

- See all sample for loop shell script [4] in our bash shell directory.
- man bash
- help for
- help {
- help break
- help continue

Updated for accuracy!

URLs in this post:

[1] Image: **http://www.cyberciti.biz/faq/category/bash-shell/**
[2] Image: **http://www.cyberciti.biz/faq/category/unix/**
[3] seq command: **http://www.cyberciti.biz/tips/how-to-generating-print-range-sequence-of-numbers.html**
[4] for loop shell script: **http://bash.cyberciti.biz/script/for-loop/**
