

Filtering Text Files Using awk



Andrew Mallett

Author and Trainer

@theurbanpenguin | www.theurbanpenguin.com



Overview



Awk fundamentals

- Why awk
- Formatting text output with awk
- Filter lastlog with awk





Why awk?



BEGIN BLOCK

Print headers

MAIN BLOCK

Print fields

END BLOCK

Summary



```
$ awk -F: '{ print $0 }' /etc/passwd  
$ awk -F: '{ print NR ": " $0 }' /etc/passwd  
$ awk -F: '/bash$/{print NR ": " $0 }' /etc/passwd
```

Main Block

Often, we can make use of the one block of code in AWK, the MAIN block



```
$ awk -F: ' /bash$/{ printf "%2d%s%s\n",NR,": ",$0 }' /etc/passwd
```

Print or Printf

For better alignment we can use `printf` from within `awk`. In this case it allows us to align the numbers correctly



Demo



Using the main code block in awk

- Using print
- Adding line numbers
- Alignment using printf





Working with the BEGIN Block




```
$ awk 'BEGIN {FS=":"; printf "%4s%20s%6s\n", "Num:", "Username", "UID"} \
/bash$/{ printf "%2d%s%20s%6d\n",NR, ": ", $1, $3}' /etc/passwd
```

BEGIN Block

Using the BEGIN block we can print headers as well as setting the FS variable. This is useful in AWK files.



```
$ awk 'BEGIN {FS=":"; printf "%4s%20s%6s\n", "Num:", "Username", "UID"; COUNT=0} \
/bash$/{ COUNT++; printf "%2d%s%20s%6d\n",COUNT, ": ", $1, $3}' /etc/passwd
```

BEGIN Block

We can also declare our own variables, here COUNT. The COUNT variable is incremented in the main block before printing



Demo



Implementing the BEGIN Block

- Adding Headers
- Setting FS variable
- Creating a row counter



```
$ awk 'BEGIN {FS=":"; printf "%4s%20s%6s\n", "Num:", "Username", "UID"; COUNT=0} \
/bash$/{ COUNT++; printf "%2d%s%20s%6d\n",COUNT, ":", $1, $3} \
END { print "We have " NR " users, of which " COUNT " use BASH" }' /etc/passwd
```

END Block

The END block can be useful for summary data. Here, we print the total users and those using BASH



Demo



END Block Summary

- Total users
- Those using BASH





AWK Files



```
# BEGIN block: This block is executed once at the beginning of the program.
BEGIN {
    print "This is the beginning of the program"
}
# MAIN block: This block is executed for each input record.
{
    # Print the first field of each input record.
    print $1
}
# END block: This block is executed once at the end of the program.
END {
    print "This is the end of the program"
}
```

Just Like sed

We can implement AWK files or script to persist are command line entries



Demo



AWK Files

- Simple awk file
- Passwd awk file





Processing Output



Lastlog Formatting

```
#We exclude certain lines from being processed
!(/Never logged in/ || /^Username/ || /^root/) {
    COUNT++;
    if ( NF == 8 )
        printf "%8s %2s %3s %4s\n", $1,$5,$4,$8;
    else
        printf "%8s %2s %3s %4s\n", $1,$6,$5,$9};
END {
    print "===== ";
    print "Total Number of Users Processed: ", COUNT;
}
```



Demo



Lastlog processing

- Lastlog command
- Lastlog awk file



Summary



Fundamentals of AWK

- AWK is a reporting tool that allows for
 - BEGIN
 - MAIN
 - END blocks
- Print/Printf
 - print is more simple
 - printf allows alignment
- Counting
 - NR
 - our own variable COUNT



Up Next:

Filter Data from Differing Text File Formats

