Open Computing ‘‘Hands-On’’: ‘‘Wizard’s Grabbag’’ Column: January 1994: Listings

Listing 1: The `!` script runs a command storing its output in a temporary file whose name is supplied on the standard output.

A. Listing of the `!` script:

```
#! /bin/sh
# !  Run command, store output in temporary file, output file name
# Author:  Jerry Peek
# Typical usage: comm `! sort file1` `! sort file2`

# Configuration constant:
TMPDIR=/var/tmp                  # Directory for temporary file

TempFile=$TMPDIR/bang$$          # Actual temporary file name

# Set trap to erase temporary file when signaled:
trap 'rm -f $TempFile; exit' 1 2 15

# If no command, report filename (to avoid more errors) and exit:
case $# in
0)  echo "Usage: `basename $0` command [args]" 1>&2
    echo $TempFile ; exit 1
    ;;
esac

# Run specified command line, store output, and report file name:
"$@" > $TempFile
echo $TempFile

# Let calling shell stop reading our output, clean up temp file:
exec >&-                        # Close stdout
(sleep 30; rm -f $TempFile)&
exit 0
```

B. A sample multistep operation:

```
$ sort file1 >/tmp/file1.sorted
$ sort file2 >/tmp/file2.sorted
$ comm /tmp/file1.sorted /tmp/file2.sorted
  ...
$ rm /tmp/file[12].sorted
$ []
```

C. Performing the same operation in one step with `!`:

```
$ comm `! sort file1` `! sort file2`
  ...
$ []
```

D. Using `!` to expand tabs in files (with four-column tab stops) before comparing files with `diff`:

```
$ diff `! expand -4 fileA` `! expand -4 fileB`
```

```
  ...
$ []
```

Listing 2: You can use a shell script to invoke the `uux` binary with the `-r` option.

A. Installing the basic shell script:

```
# cd /usr/bin
# mv uux UUX
# cat > uux
UUX -r $*
^D
# chmod 755 uux
# ls -l uux UUX
-rwxr-xr-x   1 root     other         10 Jan 26 17:33 uux
---s--x--x   1 uucp     uucp       59212 Jan  9  1991 UUX
# []
```

B. Listing of an enhanced `uux` shell script that provides some auditing capabilities:

```
#! /bin/sh
Logfile=/usr/spool/uucp/.Admin/uuxlog
# If log file is writable, then write information
# about the queued job, with the same format used in
# the SVR4 HoneyDanBer /var/uucp/.Admin/command file:
if [ -w $Logfile ]; then
        echo "\n$LOGNAME `date +'(%m/%d-%H:%M:%S)'` UUX -r $*" >>$Logfile
        # Display information from the environment:
        case "$LOGNAME" in
                uucp)   echo $UU_USER | sed 's/^/    /p' >>$Logfile ;;
                root|adm|sys)     env | sed 's/^/    /p' >>$Logfile ;;
        esac
fi
# Do the deed:
UUX -r $*
```

C. Additional `uudemon.cleanup` script lines to mail the `uux`-log file and then shuffle its name, saving one backup copy:

```
# The relevant configuration constants:
MAILTO=uucp                       # Forward to site postmaster
SPOOL=/usr/spool/uucp             # SVR4 uses VAR=/var/uucp
ADMIN=$SPOOL/.Admin               # SVR4 uses ADMIN=$VAR/.Admin
OLD=$SPOOL/.Old                   # SVR4 uses OLD=$VAR/.Old

# If there's a readable uuxlog file, mail it:
[ -r $ADMIN/uuxlog ] &&
mailx -s "`uname -n` $ADMIN/uuxlog file contents" $MAILTO <$ADMIN/uuxlog
# Now back up uuxlog file:
cp $ADMIN/uuxlog $OLD/uuxlog      # keep one backup copy
> $ADMIN/uuxlog                   # truncate the log without changing
                                  # ownership, permission modes
```

Listing 3: Revisiting the `ap` command.

A. An example that doesn't display the final value of the sequence:

```
$ ap -n 10 0 3.5 0.14
```

```
0 0.14 0.28 0.42 0.56 0.7 0.84 0.98 1.12 1.26
1.4 1.54 1.68 1.82 1.96 2.1 2.24 2.38 2.52 2.66
2.8 2.94 3.08 3.22 3.36
$ []
```

B. The original line 90:

```
90        n = (int) (1.5 + (limit-start)/step); /* nearest integer */
```

C. New versions for lines 21, 40, and 96 [Listing 1A, August 1992] that support no limit to the number of elements per line as the default display:

```
21       -n NPL: number of elements per line (default: infinity)\n\
40        npl = 0;
96        if (j == n || npl > 0 && j % npl == 0)
```

Go To Content | Search

Last Modified: Tuesday, 22-Aug-95 16:23:25 PDT