ToolMan Visualizes Disk Usage

by Daniel E. Singer
<des@cs.duke.edu>

Previously [*see the June 1997* ;login:], I discussed the qualitative issue of tracking the contents of individual directories, along with an approach (the *check* program) for addressing that problem. In this article, I discuss a more quantitative directory issue: determining where space is being consumed in a directory hierarchy; and I present a tool to assist in this process.

In my current life as a system administrator in an academic department, I'm often faced with the unenviable task of having to ask people to reduce their disk space usage. Alternatively, someone hits his disk quota limit, has trouble spotting where all those disk blocks are piled up, and comes seeking guidance (see Listing 1).

Listing 1: Over Quota (excerpt)

```
% quota -v tom
Disk quotas for tom (uid 1225):
Filesystem   usage  quota  limit
/export/home 87613  80000  100000
```

The usual approach in a UNIX environment for locating disk consumption in a directory hierarchy is the standard `du` (Disk Usage) utility. The `du` command starts at the current directory (or the directory you specify as an argument) and recursively traverses the directory hierarchy counting blocks as it goes. For each directory encountered, it prints the relative path of the directory preceded by the total block count for that directory and all its subdirectories. For a more detailed discussion of the `du` command, type `man du` for the online manual entry.

Unfortunately, `du`'s output is awfully terse and, for some, counterintuitive. Furthermore, it's not very usefully ordered or formatted. Listing 2 shows part of an example of `du` output for a smallish user directory of about 24 megabytes. The 96 lines of output for this directory have been edited down for space considerations. The `du` listing for my own home directory of around 80 megabytes is over 600 lines. (Our humble editor claims over 1,700 lines!)

Listing 2: Sample Output from `du` on a "Smallish" User Directory

```
% du ~ray
3304   /home/ray/mail
16     /home/ray/bin
20     /home/ray/calc/gs/cgs
68     /home/ray/calc/gs/mgs/OLD
16     /home/ray/calc/gs/mgs/PIV
48     /home/ray/calc/gs/mgs/ORT
154    /home/ray/calc/gs/mgs
62     /home/ray/calc/gs/qp3
56     /home/ray/calc/gs/xiaobai
294    /home/ray/calc/gs
[... 76 lines deleted ...]
2      /home/ray/eigen/v3/libdir/hp-iti
2      /home/ray/eigen/v3/libdir/hp-nuvol
2      /home/ray/eigen/v3/libdir/sgi
2      /home/ray/eigen/v3/libdir/solaris
2      /home/ray/eigen/v3/libdir/sources
2      /home/ray/eigen/v3/libdir/sun
16     /home/ray/eigen/v3/libdir
2066   /home/ray/eigen/v3
5792   /home/ray/eigen
24216  /home/ray
```

As you can see, a listing of this type can be intimidating, especially for nontechnical users and especially with larger directory trees. Asking users to run `du` is, regrettably, unlikely to fulfill their needs or yours. Clearly, a better mousetrap is needed.

Tool time! (This is where the theme music is supposed to come in. . . .) To fill this niche in disk and account management, I wrote a Bourne shell script called `duf` (`du` Formatter). It runs `du`, and sorts and formats the output such that it's easy to see where disk consumption is concentrated. Hierarchical relationships are logically and visually maintained. Listing 3 shows the output of `duf` run on our sample directory, again edited down. (Note: Perl would also be a good language for building this tool. This is left as an exercise for the reader!)

Listing 3: Sample Output from `duf`

```
% duf ~ray
/home/ray:

  12108   TOTAL (kilobytes)
```

```
    4648  tex
          1732  biblio
                  1474  stewart
                   257  bjorck
           953  pencil
                   210  calc
                         83  seg
                   121  Garbage
           559   vecpar
                   395  paper
                        173  GRAF
                               45  garbage
                   163  trasp
                        127  GRAF
    2896  eigen
[... 64 lines deleted ...]
      48  .dt
            21  sessions
                     9  current
                     9  current.old
            12  help
                    11  equin-moa-0
             3  sessionlogs
             2    types
                     1  fp_dynamic
             1  Desktop
             1    Trash
             1  appmanager
             1    icons
             1      tmp
       8   bin
       1  .wastebasket
```

The important information is now more obvious, but the listing is still long. A handy feature of `duf` is the option to specify the hierarchical *depth* (which, interestingly enough, is displayed as the *width*) of the listing. For example, a directory might go many levels deep, but you want to see a reduced listing of just the top two levels. Listing 4 shows the sample directory, using the depth option. Of course, disk blocks lower in the hierarchy are still tallied in the directories that contain them.

Listing 4: Sample Output from `duf` with Depth Set to 2

```
% duf -d 2 ~ray
/home/ray:

  12108  TOTAL (kilobytes)

    4648  tex
          1732  biblio
           953  pencil
           559  vecpar
           467  pencil2
           304  papeleo
           269  robert
           194  aartc
            88  bgs
    2896  eigen
          1033  v3
           953  v2.1
           909  v1
    1652  mail
    1628  prb
     935  for
           399  mgs
           383  cgs
     219  calc
           147  gs
            71  spec
      48  .dt
            21  sessions
            12  help
             3  sessionlogs
             2  types
             1  Desktop
             1  Trash
             1  appmanager
             1  icons
             1  tmp
       8  bin
       1  .wastebasket
```

The `duf` utility also has command line options to:

- list all files (not just directories)
- take input from a file or standard input
- sort by name instead of by size
- add visual indentation clues to aid with multi-page listings.

And `duf` will use a pager if output is to a terminal.

So there it is, a better mousetrap – a visualization tool to help with one aspect of disk management – and another example of the *tool* approach to enhancing your effectiveness. If you're interested in a copy, `duf` is available via <*http://www.cs.duke.edu/~des/scripts/duf*> or <*ftp://ftp.cs.duke.edu/pub/des/scripts/duf*>.

It's worth noting that the master tool at work here (the workbench?) is the ubiquitous Bourne shell (`sh`). It enables you to stream multiple UNIX utilities together – in this case `du`, `nawk`, and `sort` – to seamlessly act as one. Despite its shortcomings (it has a few; `bash`, the "Bourne Again" shell, addresses some of these and is worth investigating), Bourne shell is a powerful and flexible tool for scripting. ToolMan recommends it highly!