|  |  |  |  |
| --- | --- | --- | --- |
| [Wayback Machine](http://web.archive.org/web/) | |  | | --- | | Top of Form    Bottom of Form | | |
| |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | Top of Form  Bottom of Form | | |  |  |  | | --- | --- | --- | | [JUN](http://web.archive.org/web/20110628223550/http:/partmaps.org/era/unix/award.html) | Jul | [**AUG**](http://web.archive.org/web/20110830060214/http:/partmaps.org/era/unix/award.html) | | [Previous capture](http://web.archive.org/web/20110727164000/http:/partmaps.org/era/unix/award.html) | 30 | [Next capture](http://web.archive.org/web/20110811031120/http:/partmaps.org/era/unix/award.html) | | [**2010**](http://web.archive.org/web/20100729170047/http:/partmaps.org/era/unix/award.html) | 2011 | [**2012**](http://web.archive.org/web/20120828203713/http:/partmaps.org/era/unix/award.html) | | | [165 captures](http://web.archive.org/web/20110730051506*/http:/partmaps.org/era/unix/award.html)  22 Oct 06 - 20 Apr 16 |  | | | [Close](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award.html#close) [Help](http://faq.web.archive.org/) |

**Useless Use of Cat Award**

If you've been reading [comp.unix.shell](news:comp.unix.shell) or any of the related groups [(comp.unix.questions](news:comp.unix.questions) inter alia) for any amount of time, this should be a familiar topic.

I made this web page on the topic primarily so I'd have a simpler URL than one of those ghastly Deja News searches to hand to people. I've tried to reconstruct Randal's standard form letter from looking at his postings [(see end)](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award.html#forms) and added some comments of my own.

If you came here looking for material about abuse of feline animals, try [this Alta Vista search](http://web.archive.org/web/20110730051506/http:/www.altavista.com/cgi-bin/query?pg=aq&text=yes&q=title%3a%22kitty+porn%22+and+not+%28xxx+or+title%3a%22persian+kitty%22+or+title%3acorporation%29) instead.

Contents:

* Useless Use of [cat](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award.html#cat)
* Useless Use of [kill -9](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award.html#kill)
* Useless Use of [echo](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award.html#echo)
* Useless Use of [ls \*](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award.html#ls)
* Useless Use of [wc -l](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award.html#wc)
* Useless Use of [grep | awk](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award.html#grep)
* Useless Use of [Backticks](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award.html#backticks)
* Useless Use of [test](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award.html#test)
* [Assorted Other Gripes](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award.html#gripes)
* [Related Resources](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award.html#readme)
* [Reconstructions of Randal's Form Letters](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award.html#forms)

**The Useless Use of Cat Award**

The venerable [Randal L. Schwartz](http://web.archive.org/web/20110730051506/http:/www.stonehenge.com/%7Emerlyn/) hands out Useless Use of Cat Awards from time to time; you can see some recent examples in [Deja News.](http://web.archive.org/web/20110730051506/http:/search.dejanews.com/dnquery.xp?QRY=&ST=PS&DBS=1&defaultOp=AND&maxhits=25&format=terse&showsort=date&groups=&authors=&subjects=useless+%26+use+%26+cat+%26+award&fromdate=Jan+1+1970&todate=) (The subject line really says "This Week's Useless Use of Cat Award" although the postings are a lot less frequent than that nowadays). The actual award text is basically the same each time, and the ensuing discussion is usually just as uninteresting, but there are some refreshing threads there among all the flogging of this dead horse.

The oldest article Deja News finds is from 1995, but it's actually a followup to an earlier article. By Internet standards, this is thus an Ancient Tradition.

Exercise: Try to find statistically significant differences between the followups from 1995 and the ones being posted today.

(See below for a [reconstruction of the Award text.)](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award.html#uucaletter)

Briefly, here's the collected wisdom on using cat:

The purpose of cat is to concatenate (or "catenate") files. If it's only one file, concatenating it with nothing at all is a waste of time, and costs you a process.

The fact that the same thread ("but but but, I think it's cleaner / nicer / not that much of a waste / my privelege to waste processes!") springs up virtually every time the Award is posted is also Ancient Usenet Tradition.

Of course, as Heiner points out, using cat on a single file to view it from the command line is a valid use of cat (but you might be better off if you get accustomed to using less for this instead).

In a recent thread on comp.unix.shell, the following example was posted by Andreas Schwab as another Useful Use of Cat on a lone file:

{ foo; bar; cat mumble; baz } | whatever

Here, the contents of the file mumble are output to stdout after the output from the programs foo and bar, and before the output of baz. All the generated output is piped to the program whatever. (Read up on shell programming constructs if this was news to you :-)

**Other Fun Awards**

This could evolve into a good listing of "don't do that" shell programming idioms.

**Useless Use of Kill -9**

Randal also posts his [Useless Use of Kill -9 Award](http://web.archive.org/web/20110730051506/http:/search.dejanews.com/dnquery.xp?QRY=&ST=PS&DBS=1&defaultOp=AND&maxhits=25&format=terse&showsort=score&groups=&authors=&subjects=useless+%26+use+%26+kill&fromdate=Jan+1+1970&todate=) although much less frequently.

(See below for a [reconstruction of the Award text.](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award.html#uuk9letter) It explains the issues clearly enough.)

**Useless Use of echo**

This is really a special case of [Useless Use of Backticks](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award.html#backticks) but it deserves its own section because it's something you see fairly frequently.

The canonical form of this is something like

variable="something here, or perhaps even the result of backticks"

some command -options `echo $variable`

Depending a little bit on what exactly you have the variable for, this can be reduced at least to

variable="something here, or perhaps even the result of backticks"

some command -options $variable

and there is often no real reason to even think of using echo in backticks when the simpler construct will do.

(There is a twist: echo will "flatten" any whitespace in $variable into a single space -- unless you double-quote $variable, of course --, and sometimes you can legitimately use echo in backticks for this side effect. But that's rarely necessary or useful, and so most often, this is just a misguided use of echo.)

There is another example in the next section, and a longer rant about [Useless Use of Backticks](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award.html#backticks) further down the page. There is also a parallel, slightly different example on the [Backticks Example page](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award-example-backticks.html#echo-backticks)

**Useless Use of ls \***

Very clever. Usually this is seen as part of a for loop:

for f in `ls \*`; do

command "$f" # newbies will often forget the quotes, too

done

Of course, the ls is not very useful. It will just waste an extra process doing absolutely nothing. The \* glob will be expanded by the shell before ls even gets to see the file names (never mind that ls lists all files by default anyway, so naming the files you want listed is redundant here).

Here's a related but slightly more benign error (because echo is often built into the shell):

for f in `echo \*`; do

command "$f"

done

But of course the backticks are still useless, the glob itself already does the expansion of the file names. (See [Useless Use of echo](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award.html#echo) above.) What was meant here was obviously

for f in \*; do

command "$f"

done

Additionally, oftentimes the command in the loop doesn't even need to be run in a for loop, so you might be able to simplify further and say

command \*

A different issue is how to cope with a glob which expands into file names with spaces in them, but the for loop or the backticks won't help with that (and will even make things harder). The plain glob generates these file names just fine; [click here for an example.](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award-example-ls.html) See also [Useless Use of Backticks](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award.html#backticks)

Finally, as Aaron Crane points out, the result of ls \* will usually be the wrong thing if you do it in a directory with subdirectories; ls will list the contents of those directories, not just their names.

**Useless Use of wc -l**

This is my personal favorite. There is actually a whole class of "Useless Use of (something) | grep (something) | (something)" problems but this one usually manifests itself in scripts riddled by useless backticks and pretzel logic.

Anything that looks like

*something* | grep '..\*' | wc -l

can usually be rewritten like something along the lines of

*something* | grep -c . # Notice that . is better than '..\*'

or even (if all we want to do is check whether *something* produced any non-empty output lines)

*something* | grep . >/dev/null && ...

(or grep -q if your grep has that).

If *something* is reasonably coded, it might even already be setting its exit code to tell you whether it succeeded in doing what you asked it to do; in that case, all you have to check is the exit code:

*something* && ...

I used to have a really wretched example of clueless code (which I had written up completely on my own, to protect the innocent) which I've moved to a [separate page](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award-example-backticks.html) and annotated a little bit. It expands on the above and also has a bit about useless use of backticks [(q.v.)](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award.html#backticks)

Here's a contribution I got from Aaron Crane (thanks!):

grep -c can actually solve a large class of problems that grep | wc -l can't. If what interests you is the count for each of a group of files, then the only way to do it with grep | wc -l is to put a loop round it. So where I had this:

grep -c "^~h" [A-Z]\*/hmm[39]/newMacros

the naive solution using wc -l would have been

for f in [A-Z]\*/hmm[39]/newMacros; do

# or worse, for f in `ls [A-Z]\*/hmm[39]/newMacros` ...

echo -n "$f:"

# so that we know which file's results we're looking at

grep "^~h" "$f" | wc -l

# gag me with a spoon

done

and notice that we also had to fiddle to get the output in a convenient form.

**Useless Use of grep | awk and grep | sed**

Here's another one:

ps -l | grep -v '[g]rep' | awk '{print $2}'

(Of course, this is merely an example. If you have lsof it's probably a better solution to this particular problem; also the output of ps varies wildly from system to system so you might want to print something else than $2 and use completely different options to ps.)

Remember that sed and awk are glorified variants of grep. So why use grep at all?

ps -l | awk '!/[a]wk/{print $2}'

Usually you'd like the regex to be tighter than this, especially if your login might happen to include the letters grep or awk ...

True Story from Real Life: an older version of the GNATS system would think my real name was "System Operator" because it just went looking for the first occurrence of the letters e-r-a in the /etc/passwd file. (Well, actually, it thought my name was "System Era". It took me a while to figure out how it arrived at this somewhat whimsical conclusion. Incidentally, you also have to wonder why the author thought my real name was worth knowing, and if this is the right way to get that information. The end goal was to produce a template for an e-mail message -- perhaps my MUA would already know my real name, and even be able to produce nice e-mail headers for GNATS?)

**Useless Use of Backticks**

This is a sort of meta-award, or a corollary of several of the others. It merely consists of the general observation that shell scripts with lots of backticks in them are often possible to optimize and clean up a lot.

The [cluelessness example (I've put it on a separate page)](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award-example-backticks.html) contains a lot of badly chosen backticks and a somewhat longer discussion of what exactly they can accomplish, and some ideas for how to do it differently.

Obviously, backticks are a valid construction, and you can put them to good use in many shell scripts. I have simply noticed that newbie scripters often generously treat backticks as the hammer for all those nails they see.

**Dangerous Backticks**

A special idiom to pay attention to, because it's basically always wrong, is this:

for f in `cat file`; do

...

done

Apart from the classical [Useless Use of Cat](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award.html#cat) here, the backticks are outright dangerous, unless you **know** the result of the backticks is going to be less than or equal to how long a command line your shell can accept. (Actually, this is a kernel limitation. [The constant ARG\_MAX in your limits.h](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/arg-max.html) should tell you how much your own system can take. POSIX requires ARG\_MAX to be at least 4,096 bytes.)

Incidentally, this is also one of the Very Ancient Recurring Threads in [comp.unix.shell](news:comp.unix.shell) so don't make the mistake of posting anything that resembles this.

The right way to do this is

while read f; do

...

done <file

or, in cases where the command in backticks was something more complicated than a (Useless Use of) cat of a single file,

command that used to be in backticks |

while read f; do

...

done

The ARG\_MAX warning applies to other constructs than for loops too, of course. Generally, commands that look like

command `other command`

are usually safer with xargs:

other command | xargs command

The classic example is running something on each one of a number of files listed by find. An additional problem here is that find might find files whose names contain line breaks. GNU find can even cope with that, by using the null character as the file name separator.

# Note that we use -print0 instead of -print

# This is a GNU find -specific option

# Similarly, the -0 option to xargs is a GNU feature

find -options -more -options -print0 |

xargs -r0 command

With normal find, a user could create a file with a newline in its name, tricking you into, well, at least missing that file, and potentially something a lot more dangerous. (Imagine that you have a nightly cron job which runs as root and which removes old files from /tmp. Now imagine that a malicious user creates a directory in /tmp whose name ends in newline, and in that directory creates etc/ and etc/passwd. Then wait for it to grow old, or modify the inode's date stamp with touch ...)

In case something is still not obvious, the only disallowed file name characters under Unix are the null character and slash. If a file is called something like /tmp/moo<newline>/etc/passwd, normal find /tmp -print would output

/tmp/moo

/etc/passwd

and xargs would see two file names here. Changing the record separator to ASCII 0 means it's now valid for a file name to span multiple lines, so this becomes a non-issue.

**Useless Use of Test**

This was suggested by Chris Thompson in [comp.unix.shell:](news:comp.unix.shell)

All these UUOC postings remind me of one of my least favourite constructs in (Bourne, FTSOE) shell scripts:

mangle the world

if [ $? -ne 0 ] ; then

echo "Oh dear, mangling the world failed horribly" \

>&2 # shortened, stderr redirection by your humble HTMLizer

putback the world

exit 255

fi

when

if mangle the world ; then : ; else

echo "Oh dear, mangling the world failed horribly" \

>&2 # I'm still here

putback the world

exit 255

fi

seems to me much cleaner (even when the "then" clause can't be used in a more economical way, or, as with some shells, omitted).

(Something like this is also covered in the [wc -l section above.)](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/award.html#wc)

**Assorted Other Gripes**

Jon LaBadie sent me the following list (thanks!):

(I'll integrate it better with this page as soon as I have the time; I've been keeping it in my inbox for an embarrassing amount of time so I thought I'd better at least move it here where people can see it.)

Some things that bug me:

* Regular expressions used for searching (not substituting) that begin or end with '.\*'. Actually 'anything\*' or 'anything?'. If you are willing to accept "zero repetitions" of the anything, why specify it?
* Awk scripts that are basically cut unless reordering of fields is needed.
* Case conversions in comp.unix.shell (ex. how do I change my file names from UC to LC?) using tr/sed/awk/??? when some shells have builtin case conversions.
* Complex schemes to basically eliminate certain chars. For example DOS lines to UNIX lines. Sure read dos2unix(1), but using sed/awk/... when "tr -d '^M'" is all that is needed.
* Global changes to a file using sed to create a tmp file and renaming the tmp file, when an "ed(1)" here document would do fine.

Frederick J. Sena comments;

I *really* hate useless "kill -9"'s and "rm -fr"'s! After that, the next most annoying thing is people who use a useless "chmod 777" to make a file writable.

That's a good observation and another example of the same pattern; having only the heavy-duty sledgehammer in your toolbox, and breaking all the smaller nails with it.

Frederick also remarks:

I disagree with your awk/cut comment, as I often use awk for everything and cut for nothing because the syntax for awk is so much cleaner for one liners and I don't have to RTFM so much.

I'll counter that awk is overkill, and you don't need to reread the cut manual after you've read it once or twice; that's my experience. Also cut much more clearly conveys to the reader what is going on -- a small awk script certainly should not take a lot of time to decode, but if you do it *too* quickly, there might be subtle points which are easy to miss. By contrast, cut doesn't have those subtleties, for better or for worse.

Despite the looks of this embarrassing section, I do appreciate comments and additional ideas for this page. [Send me mail](http://web.archive.org/web/20110730051506/http:/www.iki.fi/era/feedback.html) with your suggestions!

**Related Resources**

Or, Other Diatribes.

* [Csh Programming Considered Harmful](http://web.archive.org/web/20110730051506/http:/language.perl.com/versus/csh.whynot) by Tom Christiansen
* [GUIs Considered Harmful](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/guis.txt) (ditto)

Additionally, you should probably be aware of the following FAQs:

* [Welcome to comp.unix.shell](http://web.archive.org/web/20110730051506/http:/www.faqs.org/faqs/unix-faq/shell/intro/)
* [Unix shell differences and how to change your shell](http://web.archive.org/web/20110730051506/http:/www.faqs.org/faqs/unix-faq/shell/shell-differences/)
* [Unix Frequently Asked Questions](http://web.archive.org/web/20110730051506/http:/www.faqs.org/faqs/unix-faq/faq/)

All of these are available in a nice browsable format from [the faqs.org archive](http://web.archive.org/web/20110730051506/http:/www.faqs.org/faqs/by-newsgroup/comp/comp.unix.shell.html) (which is where the above links will take you).

I have a [small collection of Unix links](http://web.archive.org/web/20110730051506/http:/partmaps.org/era/unix/index.html) with some more information, too.

**Useless Use of Cat Award form letter**

(Quote abomination)

And of course, if you've been following along for a week or two, you know

that this (BING!) is a Useless Use of Cat!

Rememeber, nearly all cases where you have:

cat file | some\_command and its args ...

you can rewrite it as:

<file some\_command and its args ...

and in some cases, such as this one, you can move the filename

to the arglist as in:

some\_command and its args ... file

Just another Useless Use of Usenet,

(.signature)

**Useless Use of Kill -9 form letter**

(Quote abomination)

No no no. Don't use kill -9.

It doesn't give the process a chance to cleanly:

1) shut down socket connections

2) clean up temp files

3) inform its children that it is going away

4) reset its terminal characteristics

and so on and so on and so on.

Generally, send 15, and wait a second or two, and if that doesn't

work, send 2, and if that doesn't work, send 1. If that doesn't,

REMOVE THE BINARY because the program is badly behaved!

Don't use kill -9. Don't bring out the combine harvester just to tidy

up the flower pot.

Just another Useless Use of Usenet,

(.signature)

$Id: award.prep,v 1.47 2000/12/16 09:24:38 era Exp $

)-: ylwols hguoht( depoh dah I naht retteb neve ,dekrow ti yltnerappA -- [.egap "elpoeP looC" s'ladnaR](http://web.archive.org/web/20110730051506/http:/www.stonehenge.com/merlyn/refindex.html) no dedulcni ti teg ot si egap siht fo esoprup elohw eht ,yllaer oN

era eriksson \* [home page](http://web.archive.org/web/20110730051506/http:/www.iki.fi/era/)