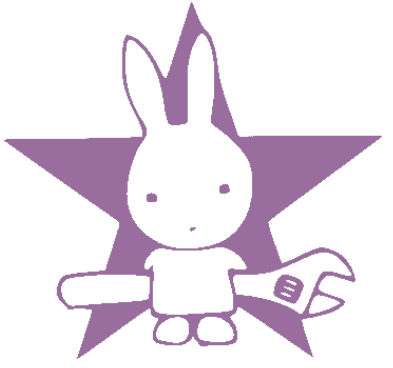
# Aus meinem Werkzeugkasten: xe

Leah Neukirchen

<leah@vuxu.org>



ASM24, München

2024-05-18

# xargs, aber in gut

„E**x**ecute for **e**ach line…“

Will man mit `xargs` einen Befehl pro Eingabezeile aufrufen, so braucht man korrekterweise:

```
xargs -n1 -d '\n' -r
```

Problem: `-d` ist nicht (POSIX-)portabel, aber wenn man `-L` verwendet gibts komische Zeilenendenfortsetzungssemantik.

Mindestens auf AIX hört `xargs` nach dem Lesen von _ auf…

Auf alten Unixen kann `xargs` auch nicht parallelisieren.

Was passiert, wenn die Eingabe leer ist?

# Mark J. Dominus runN

Zitat aus https://perl.plover.com/classes/mybin/samples/slide148.html:

> *Here's the main logic of the program and it is awesome:*
>
> ```
> my %pid;
> while (@ARGV) {
>   if (keys(%pid) < $opt{n}) {
>     $pid{spawn(@cmd, shift @ARGV)} = 1;
>   } else {
>     delete $pid{wait()};
>   }
> }
>
> 1 while wait() >= 0;
> ```
>
> *I glow with pride every time I see* `delete $pid{wait()}`.

# xe

Mein Werkzeug xe begann als Implementierung dieses Mechanismus in C.

**Zusätzliche Features:**

Lesen von Argumenten aus der Kommandozeile:

```
xe -a cc -c -- *.c
```

Parallelisierung:

```
xe -a -j8 cc -c -- *.c
```

```
xe -a -j1x cc -c -- *.c
```

# Zusätzliche Features

Shell-script-Modus:

```
xe -a -s 'convert "$1" "${1%.gif}.png"' -- *.gif
```

Pattern-Modus (experimentell…):

```
xe -a -p %.gif convert %.gif %.png -- *
```

Möglichst wenige Programmaufrufe (Standard bei `xargs`):

```
... | xe -N0 rm
```

# Sonstige Features

Bei paralleler Ausführung können Ausgaben überlappen:

```
% xe -a -j1x tjack -- 10 10 10 10
AlAAllAl wrlll work a wok and nol work and no play mrknd no play mak and n plo pakelaay
mesyakes Jas Jack a d Jull boy.  ck aack a dull b maoy.  Alkes dull bl work aoy.  All work
a JnAll work and no pnd no play makldack no play may ma a ddull bes Joay.ake  ck ake ds
Julacks JacAl a dul bok a dl worll bk ayoy.  ull boy.  .nd noAll work   pl anAlad noy ml
wake plork asAay J
makes Jack a dull workacknd no play a du
makes Jackl and no play
mallkes a bl bo dull boyy.  oy.   Jack a duAll wor.l
k and no play mal bkes Joy.  ack a d
All
```

# Gruppierung der Ausgaben

```
% xe -a -j1x -vvLL tjack -- 10 10 10 10
0001> tjack 10
0002> tjack 10
0003> tjack 10
0004> tjack 10
0003= All work and no play makes Jack a dull boy.  All work and no play makes Jack a
0001= All work and no play makes Jack a dull boy.  All work and no play makes Jack a
0004= All work and no play makes Jack a dull boy.  All work and no play makes Jack a
0002= All work and no play makes Jack a dull boy.  All work and no play makes Jack a
0001= dull boy.  All work and no play makes Jack a dull boy.  All work and no play
0002= dull boy.  All work and no play makes Jac
0002< status 0
0004= dull boy.  All work and no play makes Jack a dull boy.  Al
0001= makes
0004< status 0
0001< status 0
0003= dull boy.  All work and no play makes Jack a dull boy.  All work a
0003< status 0
```

# Ausblick

- Verfügbar unter CC0 auf `https://github.com/leahneukirchen/xe` sowie fertig paketiert für Debian, NixOS, Ubuntu, Void.

- **Noch Fragen?**

- Weitere Tools siehe `https://github.com/leahneukirchen/leahutils`