# My Tools

# My Tools

- Stand-alone
  - Or, intended for shell programming
- C programs, shell scripts
  - ANSI, POSIX (as much as possible)
- Usually take `--help` or `-man`
- Try to do something reasonable if called with no arguments

# My Tools

- st
  - status report
- note
  - notes to myself

# My Tools

- edate
  - extended date(1)
- dateops
  - date operations
- lstat
  - lstat(2) for sh
- mkstemp
  - mkstemp(3) for sh
- baseconv
  - base converter
- strcmp, strncmp
  - strcmp(3) for sh
- getman, hexify
  - built-in man pages for C programs

# st – status (report)

# st – status (report)

- Data entry
    - “What are you doing now?”
        - Interactive, command line
- Reports
    - Several forms

# st – status (report)

- Data entry
  - `st`
    - starts `${EDITOR}` on a new file
  - `st working on RT-12345`
    - writes the string to a file

# st – status (report)

- Data entry
  - defaults to today, now
  - can be made to look like entry was made yesterday
  - or any given date/time

# st – status (report)

- Reports
  - a specified day
  - this week, last week
  - a range of dates

# st – status (report)

- Reports
  - show each entry
  - can collapse all entries for a given day
- `st -w [-p] [-f]`
  - weekly report
- `st -r [date1 [date2]] [-f]`
  - report for date(s)

# st – status (report)

```
$ st -w
```

```
Tue 03 Nov 2009 09:00:00 1257256800
-----------------------------------
    workshop: advanced topics

Wed 04 Nov 2009 09:00:00 1257343200
-----------------------------------
    opening remarks, keynote (Werner)

Wed 04 Nov 2009 11:00:00 1257350400
-----------------------------------
    hallway track / recovery

Wed 04 Nov 2009 14:00:00 1257361200
-----------------------------------
    IT: advanced persistent threat
```

# st – status (report)

```
$ st -w -f
```

```
Tue 03 Nov 2009 09:00:00 1257256800
-----------------------------------
    workshop: advanced topics

Wed 04 Nov 2009 09:00:00 1257343200
-----------------------------------
    opening remarks, keynote (Werner)

    hallway track / recovery

    IT: advanced persistent threat

    IT: Google Wave
```

# st – status (report)

- Shell script
  - drop in `$HOME/bin`
  - `chmod u+x`
- Depends on:
  - `dateops`
  - `edate`
  - `mkstemp`

# note – notes to myself

# note – notes to myself

- `ln note n`
- Data entry
  - Interactive, command line, pipe
- List , print
  - Names, first lines, whole files
- Search notes
  - Uses `egrep`

# note – notes to myself

- Data entry
  - `note`
    - interactive
  - `note John wants tiddlywiki`
    - writes command line args to a file
  - `note -`
    - write stdin to a file

# note – notes to myself

- List, print
  - `note -l`
    - list all notes and first lines
  - `note -L`
    - list all full paths
  - `note -p file […]`
    - print contents of file(s)

# note – notes to myself

```
$ note -l
n.BWn10m:dimsum
n.BWoS5e:bblisa tools
n.BWoS5u:tiddlywiki
n.BWvWgC:mary’s
n.BXDJKj:atw notes
n.BXHjcA:job stuff
```

# note – notes to myself

```
$ note -p n.BWn10m
dimsum

Windsor Dimsum Cafe
China Pearl
Imperial Garden
Grand Chau Chow
```

# note – notes to myself

- Shell script
  - drop in `$HOME/bin`
  - `chmod u+x`
- Depends on:
  - `baseconv`
  - `mkstemp`
  - `[edate]`

# More Tools

# More Tools

- Other tools depend on these
  - Also useful on their own

# My Tools

- mkstemp
  - mkstemp(3) for sh
- edate
  - extended date(1)
- dateops
  - date operations
- baseconv
  - base converter
- lstat
  - lstat(2) for sh
- strcmp, strncmp
  - strcmp(3) for sh
- getman, hexify
  - built-in man pages for C programs

# mkstemp – make temp file

- Create a unique temporary file
  - Securely, too
- Shell interface to mkstemp(3)
- Usage:
  - `mkstemp template`
    - template must end with “`XXXXXX`”

# mkstemp – make temp file

```
$ mkstemp tmp-XXXXXX
tmp-aXjrsF

$ mkstemp /tmp/tmp-XXXXXX
/tmp/tmp-xPkXeu
```

# mkstemp – make temp file

- C program
  - drop in place
  - `make mkstemp`
  - move binary into `$PATH`
- May be able to substitute `mktemp(1)`
  - Not POSIX
  - Linux only?

# edate – extended date(1)

# edate – extended date(1)

- edate
  - `-t esec`
  - `%E` – day of month (`%e`) no leading space
  - `%f` – suffix (“st”, “nd”, “rd”, “th”)
  - `%s` – epoch seconds
- POSIX compliant
  - GNU date needs GNU libc

# edate – extended date(1)

```
$ edate '+%s'
1260359761

$ edate -t 1259672400
Tue Dec  1 08:00:00 EST 2009
```

# edate – extended date(1)

```
$ edate \
'+|%d|..|%e|..|%E|..|%E%f|'
|09|..| 9|..|9|..|9th|
```

# edate – extended date(1)

- C program
  - drop in place
  - `make edate`
  - move binary into `$PATH`

# dateops – date operations

# dateops – date operations

- “maketime”
  - YYYYMMDDHHMMSS -> epochseconds
- +d, -d, “yesterday”, “tomorrow”
- “lastmonth”, “nextmonth”

# dateops – date operations

```
$ dateops tomorrow
20091210
$ dateops yesterday
20091208
$ dateops +3
20091212
```

# dateops – date operations

```
$ dateops maketime 20090101
1230786000

$ edate -t 1230786000
Thu Jan  1 00:00:00 EST 2009

$ dateops maketime 20090101061500
1230808500
```

# dateops – date operations

- C program
  - drop in place
  - `make dateops`
  - move binary into `$PATH`

# baseconv – base converter

# baseconv – base converter

- Convert between bases:
  - 2, 8, 10, 16, 32, 62, 64
    - base 64 integers, ***not*** Base64 encoding

# baseconv – base converter

- “Digits”:
  - base 16: `0-9 A-Z`
  - base 32: `A-Z 0-5`
  - base 62: `A-Z a-z 0-9`
  - base 64: `A-Z a-z 0-9 + -`

# baseconv – base converter

```
$ baseconv -i 10 -o 2 9
1001

$ baseconv -i 10 -o 64 1230808500
BJXKW0
```

# baseconv – base converter

- Shell script
  - Drop in `$PATH`
  - `chmod u+x`
- Depends on `dc`
  - If not installed, available as a package for most systems
  - `bc` is part of the POSIX spec
    - but doesn’t support bases > 16

# lstat – lstat(2) for sh

# lstat – lstat(2) for sh

- Show contents of `struct stat`
- Lots of options
  - values in bases 2-16 or human-readable format
  - select contents by name
  - special handling for symlinks
  - quoting, separators, etc.

# lstat – lstat(2) for sh

```
$ lstat foo
2049;1734065;33216;1;1001;1001;\
40956;1260308736;1260302281;\
1260302285 foo

$ lstat -o 'mode:o' foo
100700 foo
```

# lstat – lstat(2) for sh

```
$ lstat -o \
'uid:s;gid:s;perm:s;mtime:z' foo
adamm;adamm;u=rwx,g-rwxs,o-rwxs;\
2009-12-08.14:58:01 foo

$ lstat --man
(lots of output)
```

# lstat – lstat(2) for sh

- C program and `.h` file
  - drop in place
  - `make lstat`
    - uses `getman`, `hexify`
    - must be in `$PATH`
    - only needed for make; stand-alone once compiled

# strcmp, strncmp – …(3) for sh

# strcmp, strncmp – …(3) for sh

- Compare two strings
- Return fixed value
  - `-1, 0, 1`
- Shell may not like `-1`
  - use `-v` flag
  - “`less`”, “`equal`”, “`greater`”

# getman, hexify

# getman, hexify

- Built-in man pages for C programs
- You don’t really care about this
  - except it’s used by lstat

# getman, hexify

- getman
  - extract man page source from .c file
  - emit C code to be included
- hexify
  - convert man page to array of hex-encoded strings

# getman, hexify

- Again, you probably don’t care
- Don’t worry (be happy)

# getman, hexify

- Write C program
- Use “`hexify -f`” to get “extra” C code
  - insert code into `.c` file
- Write man page
  - Plain ASCII, in `.c` file, surrounded by special comments
- `getman foo.c | hexify > foo.hex`
- `make foo`
- `rm foo.hex`

# After Today . . .

- Questions?
  - *`<adamm@menlo.com>`*
- Tools, slides:
  - `http://menlo.com/bblisa/`