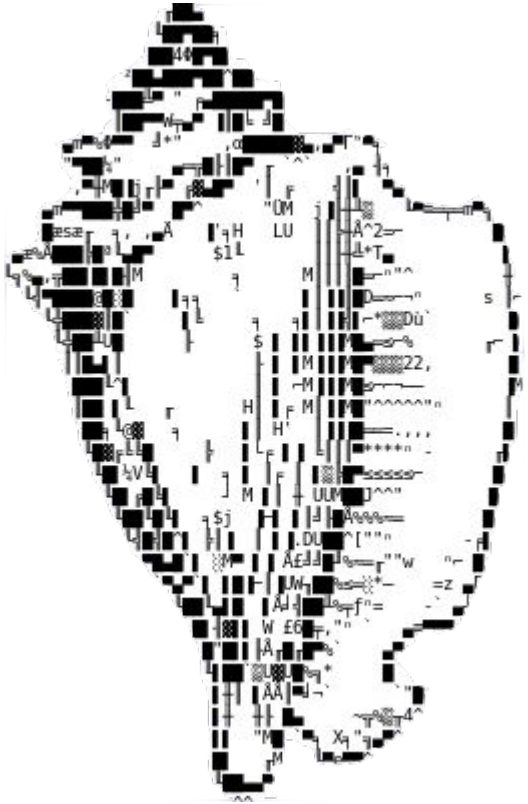


Xonsh

A Python shell
for everyday use

__init__

- Switched to xonsh full-time about a month ago
  - Spoiler: I *really* like it!
- Little time, *much* to cover
- Going to assume a rather high knowledge of both shell and Python
- If you have questions, please ask! Now or after...
  - I'd rather run long than leave unanswered questions
  - Always happy to dive deep into Unix, Python, etc.

"If you know Python and you use Bash ever, you already know why you want more Python in your shell."

- Aaron Christianson, Xonsh Quickstart

Shell is great

- Run commands - direct invocation, "builtins", and transparent fallback "path"
  - Arguments, flags, subcommands, and other "command line interface"
- Pipelines - link small commands together to do remarkable things
- Customization - helpful prompt, aliases, configurable `$PATH`
- Scripting
  - Wellll... that part really isn't great...

Shell is terrible

- Everything is strings
- Often parsed badly; problems lurk *everywhere*
  - `"Quotes" won't save you!` 🤦 is a valid filename
    - "But that's an edge-case"
    - No, not really, and our jobs would be *SO much easier* if we didn't have to deal with edge-cases!
- Lots of commands are arcane (and archaic)
  - What are the differences between `sed`, `grep`, and `egrep` regular expressions, hmm?
- Syntax is a mess

Don’t be like me:

```
: ${GetChunkDelim:="###"}
if [ "$1" == "-h" -o "$1" == "--help" -o -z "$1" -o -z "$2"
]; then
  cat <<USAGE
usage:...
USAGE
  exit 0
fi
cat "$1" | sed -e"1,/^${GetChunkDelim}[     ][  ]*$2/ d"
-e"/^${GetChunkDelim}/,$ d"
```

Python is great

- Real data structures
- Real control structures
- Lots of powerful libraries

...but not good at shell things

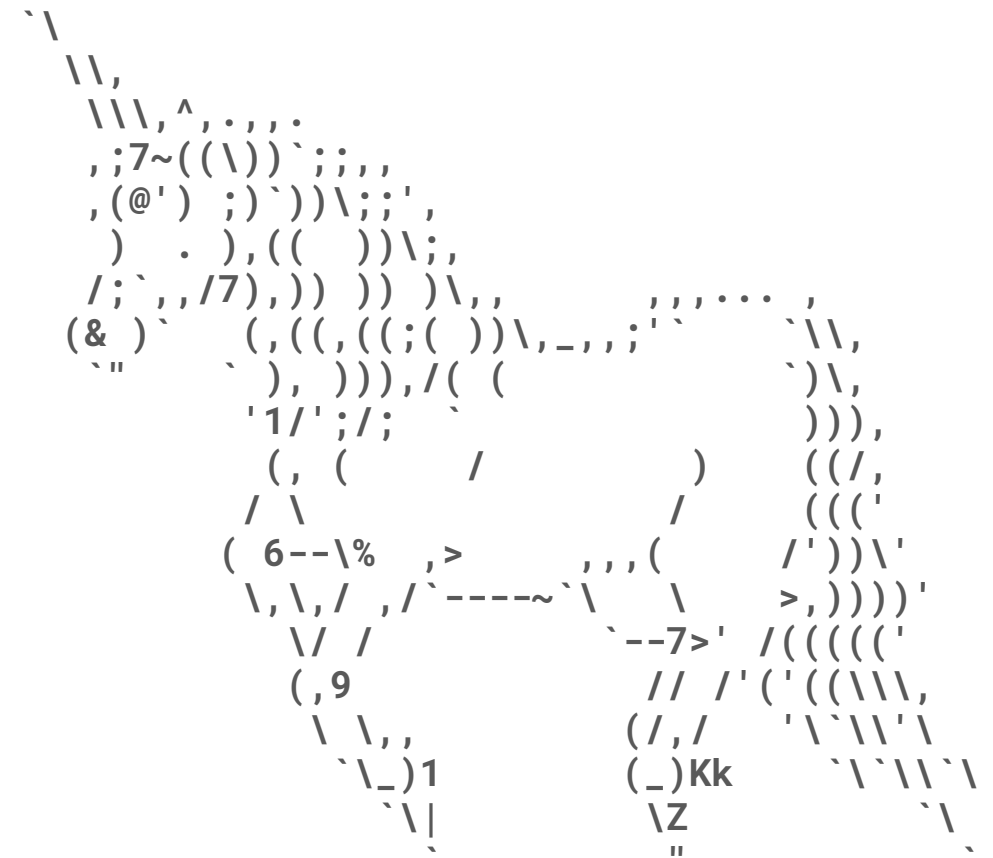
Pipe vs. subprocess

```
# $ dmesg | grep hda
p1 = Popen(["dmesg"], stdout=PIPE)
p2 = Popen(["grep", "hda"], stdin=p1.stdout, stdout=PIPE)
p1.stdout.close()  # Allow p1 to receive a SIGPIPE if p2
exits.
output = p2.communicate()[0]
```

Why not both?

Imagine an environment with...

- Simple, direct, command invocation, with all your old shell friends
- Pipes, redirection, job control
- Full Python interpreter
- Fancy prompts, completions, extensions
- Multi-line editing



Xonsh

- Pronounced like 'K'; spelled with an 'X' 'cos Xes are cool
- Brings together Python and shell (aka "subprocess mode")
- With ways to work with both and move data between the two
- Plus loads of modern terminal bells & whistles, customizability, etc.

Brief tour

- Per-line "duck typed" modes
  - if it parses like Python... (otherwise "subprocess mode")
- Various extra syntax to force or mix modes:
  - `@()` for python, `$()`, `!()`, `$[]`, `![]` for shell
- `$VAR` for global environment variables, accessible from either shell or Python
  - Python types get `str`-ified in subprocess mode
  - Special rules to split / join `$PATH`-like vars
- `aliases` is a dict and can map to strings, lists, or callables
  - powerful, but a bit tricky
- "glob expressions", using backticks, are super-cool; "p-strings" (and p-globs) produce Path objects

What's it like?

- Mostly "just works" - type shell commands and pipelines or python at any prompt anytime
- Best of both worlds: great shell features, and WAY better than the standard Python interpreter
- Dual-mode works decently well, like "house with attached garage/basement"
- Configuration takes some learning and experimentation
- Interaction between shell and Python *definitely* takes some learning and experimentation, just starting to get my "sea-legs"
- Barely scratched the surface, ask me again in a year

Demo

Gotchas

- Not recommended as $SHELL - set in terminal apps instead
- Backticks are different than sh/bash, use `$( )`
- Backslashes don't work for character escaping in shell, use quotes
- Must use `env` for setting "temporary" env vars
- Anaconda, standard virtualenv tools, and similar "`source`"-based things expect specific mainstream shells; use `vox` / `autovox`, for example
- Still kinda new; some rough spots, documentation is thin in places
  - *Contribute!*
- To use pipes in aliases wrap in callable

Join me!

- Radically different and, I think, MUCH better
- Build deeper familiarity with Python instead of piles of commands that aren't very good for other larger things
- Easy to try, just install and run
- Grow your ideas from hacks into proper tools
- Environment is written in Python, fix, change, document, and contribute to it!

Getting Started Links

- Talks & articles
  - Check out the PyCon talk for a whirlwind tour; somewhat dated, xonsh has improved significantly since then, but still impressive and entertaining
- Xonsh installation
  - Lots of ways to install, pick your poison ¯_(ツ)_/¯
- Xonsh quickstart
  - *Really* good tutorial, maybe better than the official one
  - While you're in the neighborhood, this is also really good:
    - Replacing Bash Scripting with Python
- Xonsh cheatsheet
  - Keep this handy when you're learning the different modes / substitutions / extensions to navigate the choppy waters where Python and Shell mix
- Official tutorial
  - Definitely read this and refer to it, but I've found the previous two links more useful "in the moment"

Other Resources

- Home page
  - Lots more documentation here, keep diging as you get more familiar or you find you have specific needs
- Source code
  - The great thing about a shell written in the language it's based on is that you can read the code to understand how it really works - and contribute back!
- Virtual Environments
  - Virtualenv discussion