Toolsmithing

convenience through code

Jeremy Thurgood

PyConZA: 6 October, 2023

What is a toolsmith?

A person who makes tools

What is a toolsmith?

A person who makes tools

More specifically, in the context of this presentation:
toolsmiths build tools to improve their own productivity,
and that of their friends and colleagues.

Why build tools?

Why build tools?

- Save time

Why build tools?

- Save time
- Reduce complexity

Why build tools?

- Save time
- Reduce complexity
- Existing tools don't fit

Why build tools?

- Save time
- Reduce complexity
- Existing tools don't fit
- It's fun

Save time

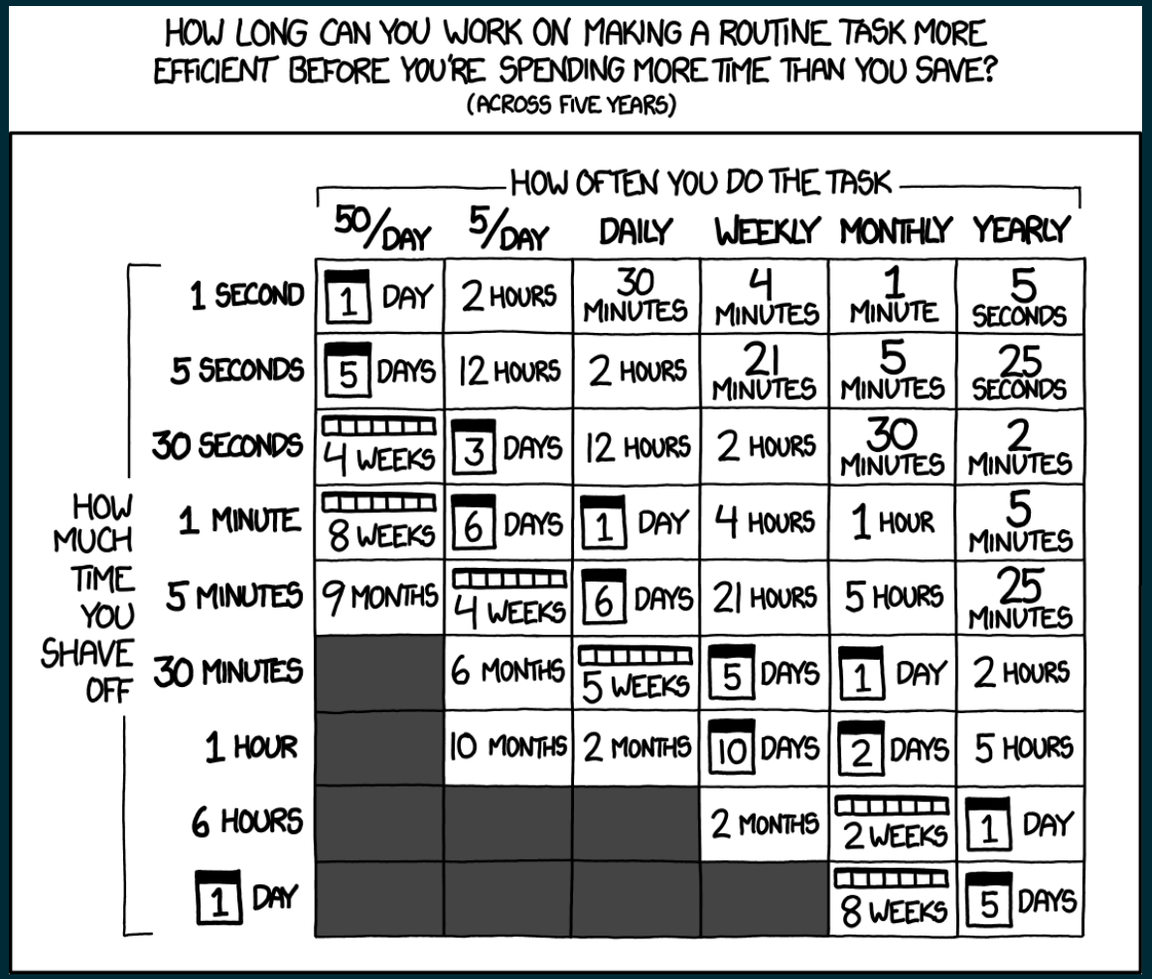

Figure 1: https://xkcd.com/1205/

Save time?

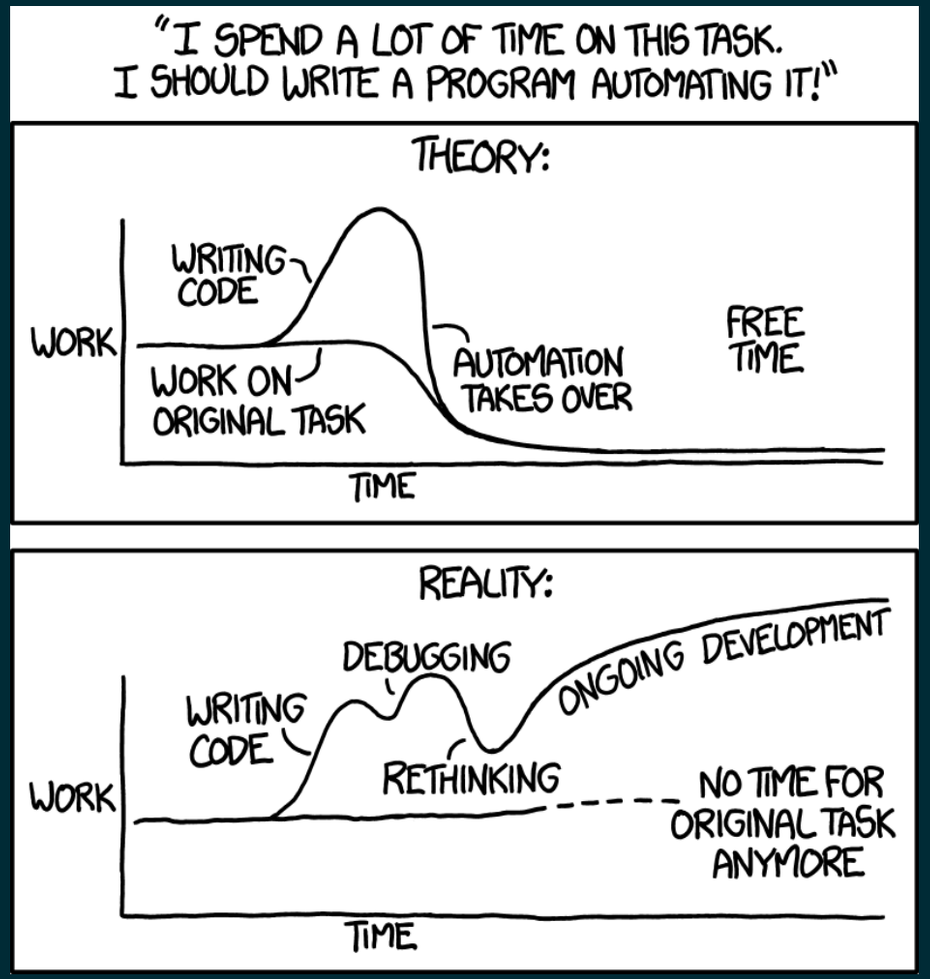

Figure 2: https://xkcd.com/1319/

Reduce complexity

Simple is better than complex

Reduce complexity

Simple is better than complex

- Get rid of boilerplate

Reduce complexity

Simple is better than complex

- Get rid of boilerplate
- Focus on the things you care about

Reduce complexity

Simple is better than complex

- Get rid of boilerplate
- Focus on the things you care about
- Remove sources of error

Existing tools

The world is full of tools

Usually you can find one to meet your needs

Existing tools

The world is full of tools

Usually you can find one to meet your needs

Maybe you can get by with one that doesn't quite fit

Existing tools

The world is full of tools

Usually you can find one to meet your needs

Maybe you can get by with one that doesn't quite fit

Or you can build your own

Fun

Happy people are productive people

Fun

Happy people are productive people

We enjoy creative work, not tedious repetition

Fun

Happy people are productive people

We enjoy creative work, not tedious repetition

Toolsmithing is creative work that replaces tedious repetition

Examples

Some tools I've built recently

- `bigterm.sh`
- `kustom-tool`
- `filter_plan_2.py`
- `clothsim.py`

`bigterm.sh`

Makes my terminal big

`bigterm.sh` » problem

Why do I need a tool for this?

`bigterm.sh` » problem

Why do I need a tool for this?

- 80x25 isn't always enough

`bigterm.sh` » problem

Why do I need a tool for this?

- 80x25 isn't always enough
- But it's a pretty good default for me

`bigterm.sh` » problem

Why do I need a tool for this?

- 80x25 isn't always enough
- But it's a pretty good default for me
- Manual resizing is inconsistent

`bigterm.sh` » problem

Why do I need a tool for this?

- 80x25 isn't always enough
- But it's a pretty good default for me
- Manual resizing is inconsistent
- Manual resizing is *annoying*

bigterm.sh » implementation

Here's the code in its entirety:

```
1 #!/bin/bash
2
3 # See the CSI section at https://www.xfree86.org/current/ctlseqs.html
4
5 printf '\e[8;50;195t'  # resize to 195x50 characters
6 printf '\e[3;200;0t'   # move to position (200, 0) pixels
```

`bigterm.sh` » results

How did it work out for me?

`bigterm.sh` » results

How did it work out for me?

- I only ever notice it on computers that don't have it

`bigterm.sh` » results

How did it work out for me?

- I only ever notice it on computers that don't have it
- Six point three bazillion context switches avoided

`bigterm.sh` » results

How did it work out for me?

- I only ever notice it on computers that don't have it
- Six point three bazillion context switches avoided
- Paid for itself in happiness the first time I used it

`bigterm.sh` » lessons

What can we learn from this?

`bigterm.sh` » lessons

What can we learn from this?

A tiny tool that solves a tiny problem can still be a big win

`kustom-tool`

Cluster configuration manager

`kustom-tool` » problem

Why do we need a tool for this?

`kustom-tool` » problem

Why do we need a tool for this?

- Have you *seen* k8s YAML?

`kustom-tool` » problem

Why do we need a tool for this?

- Have you *seen* k8s YAML?
- Deploying other people's stuff is boring (and I don't want to do it)

`kustom-tool` » problem

Why do we need a tool for this?

- Have you *seen* k8s YAML?
- Deploying other people's stuff is boring (and I don't want to do it)
- Waiting for me to deploy stuff is slow (and they don't want to do it)

`kustom-tool` » problem

Why do we need a tool for this?

- Have you *seen* k8s YAML?
- Deploying other people's stuff is boring (and I don't want to do it)
- Waiting for me to deploy stuff is slow (and they don't want to do it)
- Focus on what's important

`kustom-tool` » requirements

What does this tool need to do?

`kustom-tool` » requirements

What does this tool need to do?

- Static deploy manifests in the repo

`kustom-tool` » requirements

What does this tool need to do?

- Static deploy manifests in the repo
- Minimal boilerplate

`kustom-tool` » requirements

What does this tool need to do?

- Static deploy manifests in the repo
- Minimal boilerplate
- Use upstream sources where practical

`kustom-tool` » requirements

What does this tool need to do?

- Static deploy manifests in the repo
- Minimal boilerplate
- Use upstream sources where practical
- Reasonably fast

`kustom-tool` » approach

How did we go about building it?

`kustom-tool` » approach

How did we go about building it?

- Find existing tools we can use
  - `kustomize`, `helm`, `ytt`, etc.

`kustom-tool` » approach

How did we go about building it?

- Find existing tools we can use
  - `kustomize`, `helm`, `ytt`, etc.
- Glue those tools together

`kustom-tool` » approach

How did we go about building it?

- Find existing tools we can use
  - `kustomize`, `helm`, `ytt`, etc.
- Glue those tools together
- Build the missing pieces

`kustom-tool` » design

Two stage operation:

`kustom-tool` » design

Two stage operation:

- `sources`: Fetch upstream sources

`kustom-tool` » design

Two stage operation:

- `sources`: Fetch upstream sources
  - k8s manifest, archive, git, helm chart, kustomize

`kustom-tool` » design

Two stage operation:

- `sources`: Fetch upstream sources
  - k8s manifest, archive, git, helm chart, kustomize
  - Fetches everything required to generate output

`kustom-tool` » design

Two stage operation:

- `sources`: Fetch upstream sources
  - k8s manifest, archive, git, helm chart, kustomize
  - Fetches everything required to generate output
  - Only required when sources have changed

`kustom-tool` » design

Two stage operation:

- `sources`: Fetch upstream sources
  - k8s manifest, archive, git, helm chart, kustomize
  - Fetches everything required to generate output
  - Only required when sources have changed
- `regenerate`: Build deployment manifests

`kustom-tool` » design

Two stage operation:

- `sources`: Fetch upstream sources
  - k8s manifest, archive, git, helm chart, kustomize
  - Fetches everything required to generate output
  - Only required when sources have changed
- `regenerate`: Build deployment manifests
  - Calls `kustomize` and `ytt` to build output

`kustom-tool` » implementation

`kustom-tool` » implementation

- Python (obviously)

`kustom-tool` » implementation

- Python (obviously)
- `trio` for concurrency

`kustom-tool` » implementation

- Python (obviously)
- `trio` for concurrency
- `mypy` and `ruff` to find most of the bugs

`kustom-tool` » implementation

- Python (obviously)
- `trio` for concurrency
- `mypy` and `ruff` to find most of the bugs
- `kustomize` plugin to call `ytt`

`kustom-tool` » implementation

- Python (obviously)
- `trio` for concurrency
- `mypy` and `ruff` to find most of the bugs
- `kustomize` plugin to call `ytt`
- 2-4 months (part time) to build

`kustom-tool` » implementation

- Python (obviously)
- `trio` for concurrency
- `mypy` and `ruff` to find most of the bugs
- `kustomize` plugin to call `ytt`
- 2-4 months (part time) to build
- It's "just worked" long enough that I forgot the rest

`kustom-tool` » results

How did it work out for us?

`kustom-tool` » results

How did it work out for us?

- Deploys take minutes to prepare instead of hours/days

`kustom-tool` » results

How did it work out for us?

- Deploys take minutes to prepare instead of hours/days
- Dev teams do 90% of the deploys themselves

`kustom-tool` » results

How did it work out for us?

- Deploys take minutes to prepare instead of hours/days
- Dev teams do 90% of the deploys themselves
- No YAML-related injuries since 2021

`kustom-tool` » results

How did it work out for us?

- Deploys take minutes to prepare instead of hours/days
- Dev teams do 90% of the deploys themselves
- No YAML-related injuries since 2021
- External tools make setup somewhat annoying

`kustom-tool` » lessons

What can we learn from this?

`kustom-tool` » lessons

What can we learn from this?

Sometimes the tool we build is essentially a production system and should be treated as such

`kustom-tool` » lessons

What can we learn from this?

Sometimes the tool we build is essentially a production system and should be treated as such

Target audience matters, especially if it's more than just you

`filter_plan_2.py`

Terraform plan filter

`filter_plan_2.py` » problem

Why do we need a tool for this?

`filter_plan_2.py` » problem

Why do we need a tool for this?

- Eight (or maybe ten?) k8s clusters to upgrade

`filter_plan_2.py` » problem

Why do we need a tool for this?

- Eight (or maybe ten?) k8s clusters to upgrade
- Terraform module update with **lots** of changes

`filter_plan_2.py` » problem

Why do we need a tool for this?

- Eight (or maybe ten?) k8s clusters to upgrade
- Terraform module update with **lots** of changes
- 1k+ lines of terraform plan output

`filter_plan_2.py` » problem

Why do we need a tool for this?

- Eight (or maybe ten?) k8s clusters to upgrade
- Terraform module update with **lots** of changes
- 1k+ lines of terraform plan output
- 100+ affected resources

`filter_plan_2.py` » problem

Why do we need a tool for this?

- Eight (or maybe ten?) k8s clusters to upgrade
- Terraform module update with **lots** of changes
- 1k+ lines of terraform plan output
- 100+ affected resources
- At least one resource that **must** be fixed

filter_plan_2.py » sample input

```
# module.cluster.aws_elasticache_cluster.redis["app-redis"] will be updated in-place
~ resource "aws_elasticache_cluster" "redis" {
      id                         = "app-redis"
    ~ security_group_ids         = [
        - "sg-03fbe4e8047f95e56",
      ] -> (known after apply)
      tags                       = {}
      # (21 unchanged attributes hidden)
  }

# module.cluster.module.eks.kubernetes_config_map.aws_auth[0] will be destroyed
# (because index [0] is out of range for count)
- resource "kubernetes_config_map" "aws_auth" {
    - binary_data = {} -> null
    - data        = {
        - "mapAccounts" = jsonencode([])
```

`filter_plan_2.py` » design

Meh, who has time to design this

`filter_plan_2.py` » design

Meh, who has time to design this

It's just a bunch of substring matches, right?

`filter_plan_2.py` » implementation

`filter_plan_2.py` » implementation

- Basically just a bunch of substring matches

`filter_plan_2.py` » implementation

- Basically just a bunch of substring matches
- Grouped into sections:

`filter_plan_2.py` » implementation

- Basically just a bunch of substring matches
- Grouped into sections:
  - EXPECTED: Displayed for completeness

`filter_plan_2.py` » implementation

- Basically just a bunch of substring matches
- Grouped into sections:
  - EXPECTED: Displayed for completeness
  - UNEXPECTED: Check these to make sure they're okay

`filter_plan_2.py` » implementation

- Basically just a bunch of substring matches
- Grouped into sections:
  - EXPECTED: Displayed for completeness
  - UNEXPECTED: Check these to make sure they're okay
  - WARN: Things that need manual fixing

`filter_plan_2.py` » implementation

- Basically just a bunch of substring matches
- Grouped into sections:
  - EXPECTED: Displayed for completeness
  - UNEXPECTED: Check these to make sure they're okay
  - WARN: Things that need manual fixing
- 200 lines of very hacky code, no tests, no repo

`filter_plan_2.py` » results

How did it work out for us?

`filter_plan_2.py` » results

How did it work out for us?

- No clusters were harmed during the upgrades

`filter_plan_2.py` » results

How did it work out for us?

- No clusters were harmed during the upgrades
- (At least, not from anything related to this)

`filter_plan_2.py` » results

How did it work out for us?

- No clusters were harmed during the upgrades
- (At least, not from anything related to this)
- Spent a couple of hours working on a better version
  - Configuration was messy and difficult
  - Not worth the effort, make a new one next time

`filter_plan_2.py` » lessons

What can we learn from this?

`filter_plan_2.py` » lessons

What can we learn from this?

A tool that you use once (or eight to ten times) and then throw away can still be valuable

`filter_plan_2.py` » lessons

What can we learn from this?

A tool that you use once (or eight to ten times) and then throw away can still be valuable

A hacky prototype that gets the job done may be all you need

`clothsim.py`

Blender cloth simulation wrangler

`clothsim.py` » problem

Why do I need a tool for this?

`clothsim.py` » problem

Why do I need a tool for this?

- I'm making a comic, which means lots of renders

`clothsim.py` » problem

Why do I need a tool for this?

- I'm making a comic, which means lots of renders
- Every panel needs cloth simulation:

`clothsim.py` » problem

Why do I need a tool for this?

- I'm making a comic, which means lots of renders
- Every panel needs cloth simulation:
  - Animate from default pose to desired pose

`clothsim.py` » problem

Why do I need a tool for this?

- I'm making a comic, which means lots of renders
- Every panel needs cloth simulation:
  - Animate from default pose to desired pose
  - Save simulation results as shape keys

`clothsim.py` » problem

Why do I need a tool for this?

- I'm making a comic, which means lots of renders
- Every panel needs cloth simulation:
  - Animate from default pose to desired pose
  - Save simulation results as shape keys
  - Apply shape keys and clear simulation data

`clothsim.py` » art (unsimulated)

Figure 3: Unsimulated

`clothsim.py` » art (simulated)

Figure 4: Simulated

`clothsim.py` » implementation

`clothsim.py` » implementation

- The same as the manual process, but automated

`clothsim.py` » implementation

- The same as the manual process, but automated
- Blender's APIs are made of global mutable state

`clothsim.py` » implementation

- The same as the manual process, but automated
- Blender's APIs are made of global mutable state
  - … because that's what 3d modelling is

`clothsim.py` » implementation

- The same as the manual process, but automated
- Blender's APIs are made of global mutable state
  - … because that's what 3d modelling is
  - I still don't like it much, though

`clothsim.py` » implementation

- The same as the manual process, but automated
- Blender's APIs are made of global mutable state
  - … because that's what 3d modelling is
  - I still don't like it much, though
- Very incomplete, but usable

`clothsim.py` » results

How did it work out for me?

`clothsim.py` » results

How did it work out for me?

Dunno yet, art time has been limited

`clothsim.py` » results

How did it work out for me?

Dunno yet, art time has been limited

Early results are promising, though

`clothsim.py` » lessons

What can we learn from this?

`clothsim.py` » lessons

What can we learn from this?

Some tools will continue to evolve forever as requirements change

`clothsim.py` » lessons

What can we learn from this?

Some tools will continue to evolve forever as requirements change

If you try hard enough, you can find a way to show off your artwork at a tech conference

`clothsim.py` » lessons

What can we learn from this?

Some tools will continue to evolve forever as requirements change

If you try hard enough, you can find a way to show off your artwork at a tech conference

Conclusion » summary

What did we learn?

- `bigterm.sh` » a tiny tool can still be a big win
- `kustom-tool` » it might be a core part of your infrastructure
- `filter_plan_2.py` » value doesn't require longevity
- `clothsim.py` » it's okay to change it every time you use it

Conclusion » thoughts and advice

So, you want to start building tools…

Conclusion » thoughts and advice

So, you want to start building tools...

- Don't be afraid to try

Conclusion » thoughts and advice

So, you want to start building tools...

- Don't be afraid to try
  - You'll almost certainly learn something, at least

Conclusion » thoughts and advice

So, you want to start building tools...

- Don't be afraid to try
  - You'll almost certainly learn something, at least
- Don't be afraid to fail

Conclusion » thoughts and advice

So, you want to start building tools...

- Don't be afraid to try
  - You'll almost certainly learn something, at least
- Don't be afraid to fail
  - Sometimes it's not worth the effort, and that's okay

Conclusion » thoughts and advice

So, you want to start building tools...

- Don't be afraid to try
  - You'll almost certainly learn something, at least
- Don't be afraid to fail
  - Sometimes it's not worth the effort, and that's okay
- There's more than one way to do it

Conclusion » thoughts and advice

So, you want to start building tools...

- Don't be afraid to try
  - You'll almost certainly learn something, at least
- Don't be afraid to fail
  - Sometimes it's not worth the effort, and that's okay
- There's more than one way to do it
  - Pick the one that fits the problem you're solving

That's all, folks

Questions?