

The best programmer I know

Daniel Terhorst-North

@tastapod@mas.to

@tastapod

Part 1: Getting the job done

Just start!



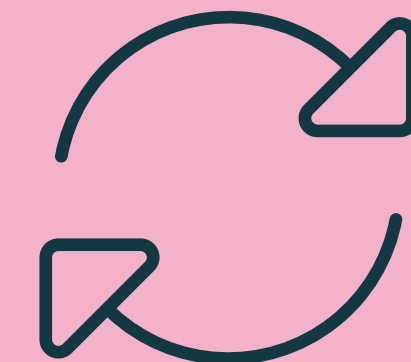
Resist procrastinating

- *Start anywhere*
- *Seriously, anywhere!*
- *Doing is researching*



Know you don't know

- *It doesn't have to be right, or even good*
- *You will rewrite it!*



Iterate wildly

- *'Stoplight to stoplight'*
- *Try, fail, learn, repeat*

Build a product



Invest in the outcome

- *Code is just the means*
- *Have no emotional attachment to it!*



Study the domain

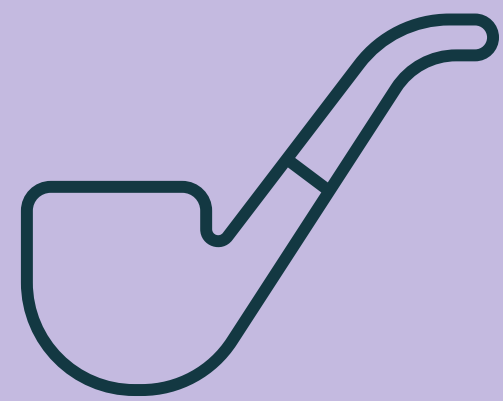
- *and its inhabitants*
- *Represent this in code*



Watch your users

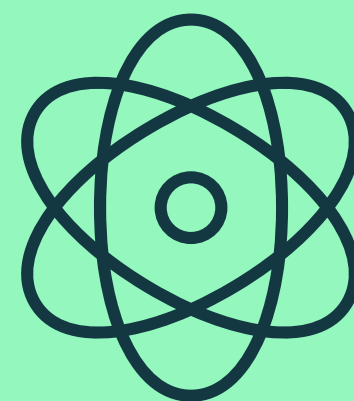
- *What frustrates them?*
- *Simplify it; eliminate it*

Solve for *now*



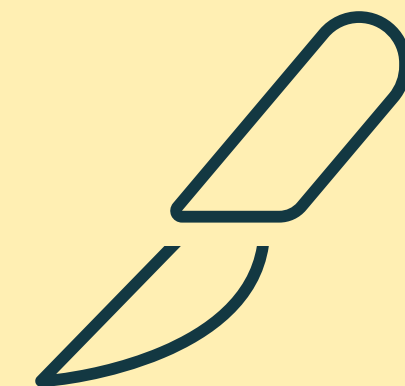
Learn to see what is really there

- *not what you are conditioned to see*
- *Develop 'first sight'*



Solve the real problem

- *not some fancy generalised version*
- *Caution: may reduced dependencies*



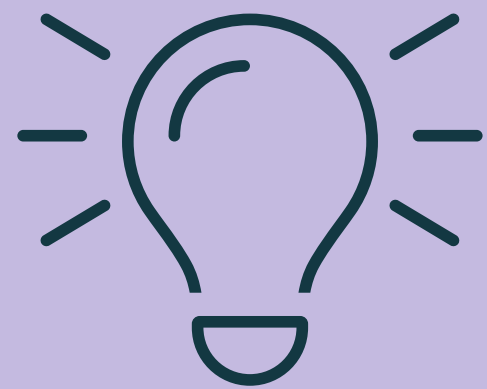
Strive for simplicity

- *'the simplicity the other side of complexity'*
- *Write the README, then reduce the embarrassment!*

Part 2: Choosing the right tool

Choosing the right tool

...for the product, *not* the team!



Teams can learn!

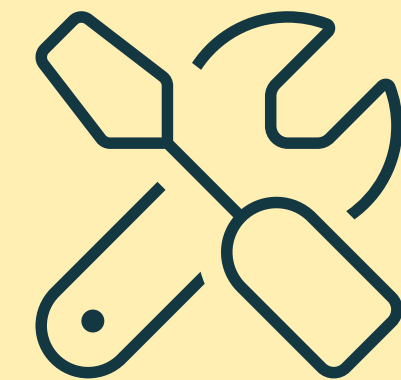
- *Code outlive teams*
- *Data outlives code, and organisations!*
- *Be kind to future you*



Do the simplest thing, not the easiest

- *h/t Rich Hickey**
- *Minimise distance to solution space*

*<https://dannorth.net/go/simple-made-easy>



The 'right tool' may change over time

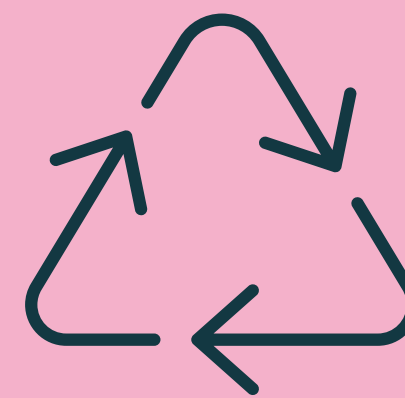
- *so be prepared to revisit your choices*
- *'Make the change easy'*

Make the change easy



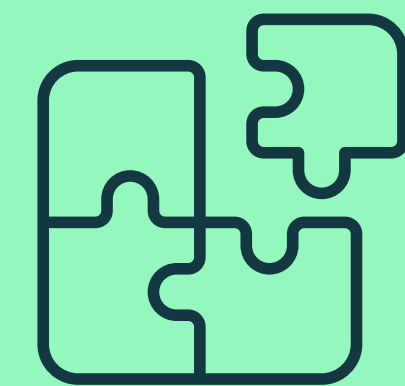
Minimise blast radius

- *Write small, self-contained 'hacks'*
- *Spike and Stabilise*
- *'Try now, pay later'*



Reduce, reuse, recycle

- *Easy to decompose*
- *Easy to restructure*
- *Easy to rewrite*



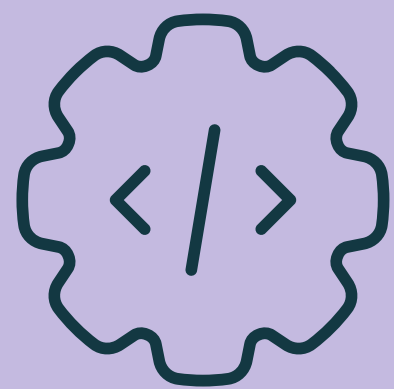
Then do the same with production code!

- *CUPID* is your friend*
- *Architecture as options*

* <https://cupid.dev>

Choosing the right tool

Be a polyglot



Explore languages, tools, paradigms

- They will give you different perspectives
- Try Advent of Code*

* <https://adventofcode.com>



Be 'full-stack'

- What makes a great web page?
- a great service API?
- a great architecture?



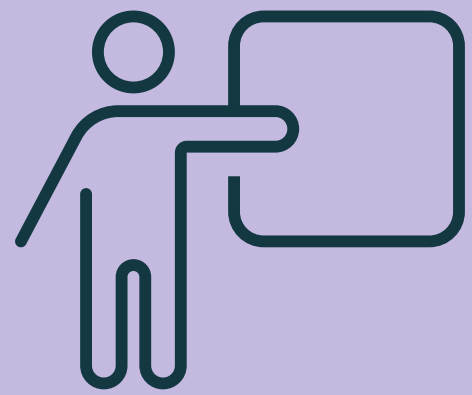
Be *really* full-stack

- Redesign the process
- Use hardware!
- Challenge the premise

Part 3: Caring about the team

Caring about

...others



Find joy in helping others learn

- *Teaching is the best way to learn*
- *Sometimes just encouraging words*



Send the team home!

- *No one should be working late*
- *A rested team is an effective team*



Be kind

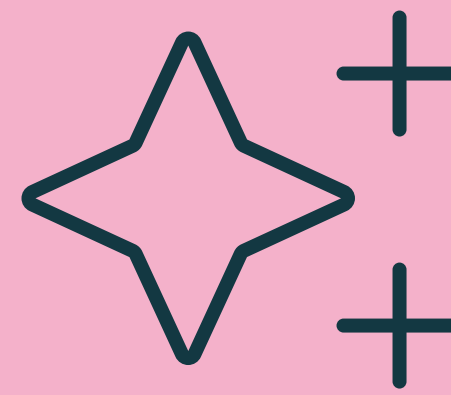
- *Assume everyone is doing their best*
- *Build psychological safety*

...staying current



Join communities

- *Be a net contributor*
- *Find people who will challenge you*
- *Throw the net wide*



Try new things

- *But retain a healthy scepticism*
- *Especially about blockchain 'AI'*



Practise, practise, practise

- *There is no 'innate programming gift'*
- *Be prepared to be rubbish at new things!*

Caring about

...yourself



Have interests outside of programming

- *Find a sport or activity*
- *If you don't have a thing, try lots of things!*



Go home on time

- *Who will remember you working late?*
- *Sleep is the best debugger*



Be kind to yourself

- *You've got this*
- *'Yesterday I was wrong'*

Wrapping up

- ◉ Get the job done
- ◉ Choose the right tool
- ◉ Care about the team

Wrapping up

Above all, be kind

Thank you

dannorth.net

@tastapod@mas.to

linkedin.com/in/tastapod