Good morning

“from banners to apps” = lessons from my history

(but not strictly true, banners weren't around in 1995)

1: Nostalgia.

2: Stuff that everyone should already know (and probably does).

3: Where I stole this stuff from.

1995: Office admin job > evening classes (C++) > Search for work.

Tiny web industry.

Sites look ugly & clunky now – but were innovative for the time.

So many limitations, so few limits.

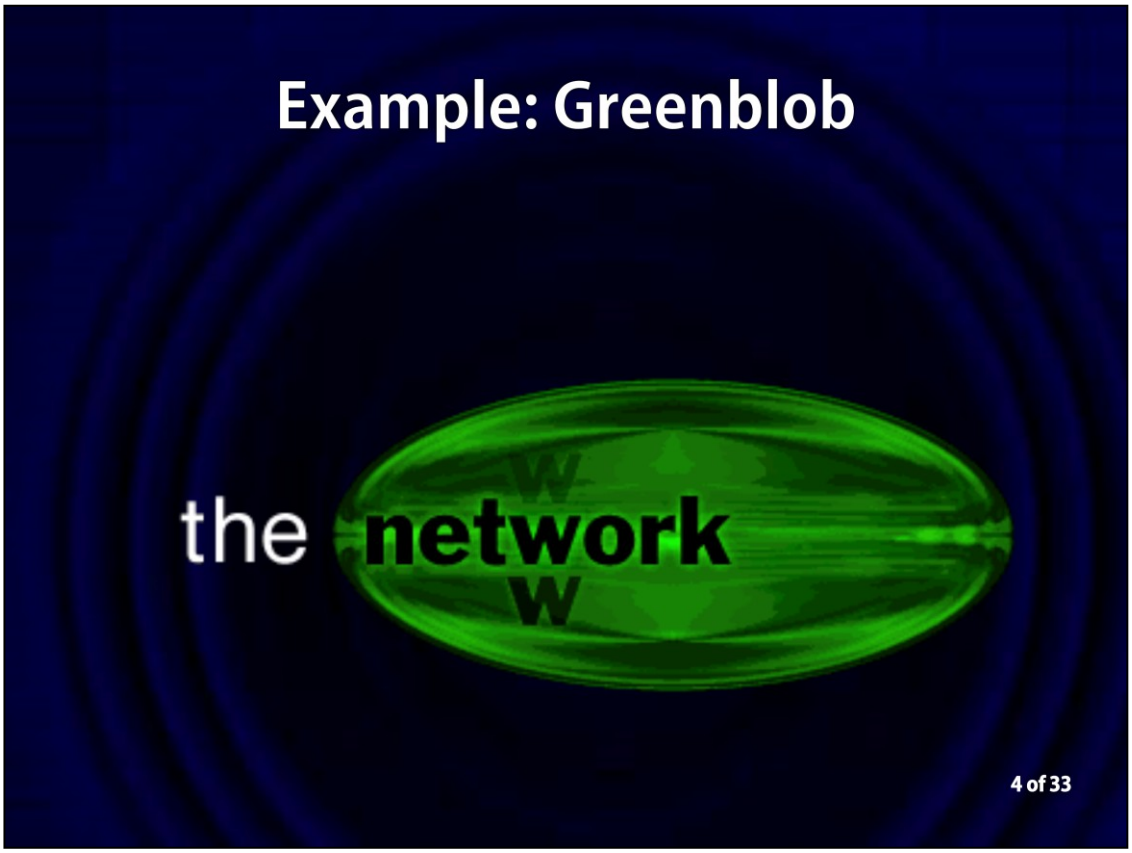
Being pushed and finding creative ways to work around those limits.

Diesel, Jolly Rancher, CompuServe, Gspot, O&M.

First ever UK web awards: Yell/.net award for Diesel.

Technical: learnt Perl in a night, wrote an HTML pre-processor in Perl. Wrote a CMS in Perl.

WORK CAN BE FUN!



HTML pre-processor (precursor to ASP/PHP/JSP)

Customisable site

Animation & sound

Future Splash vs. Liquid Motion

1998 buy-out of Hard Reality.

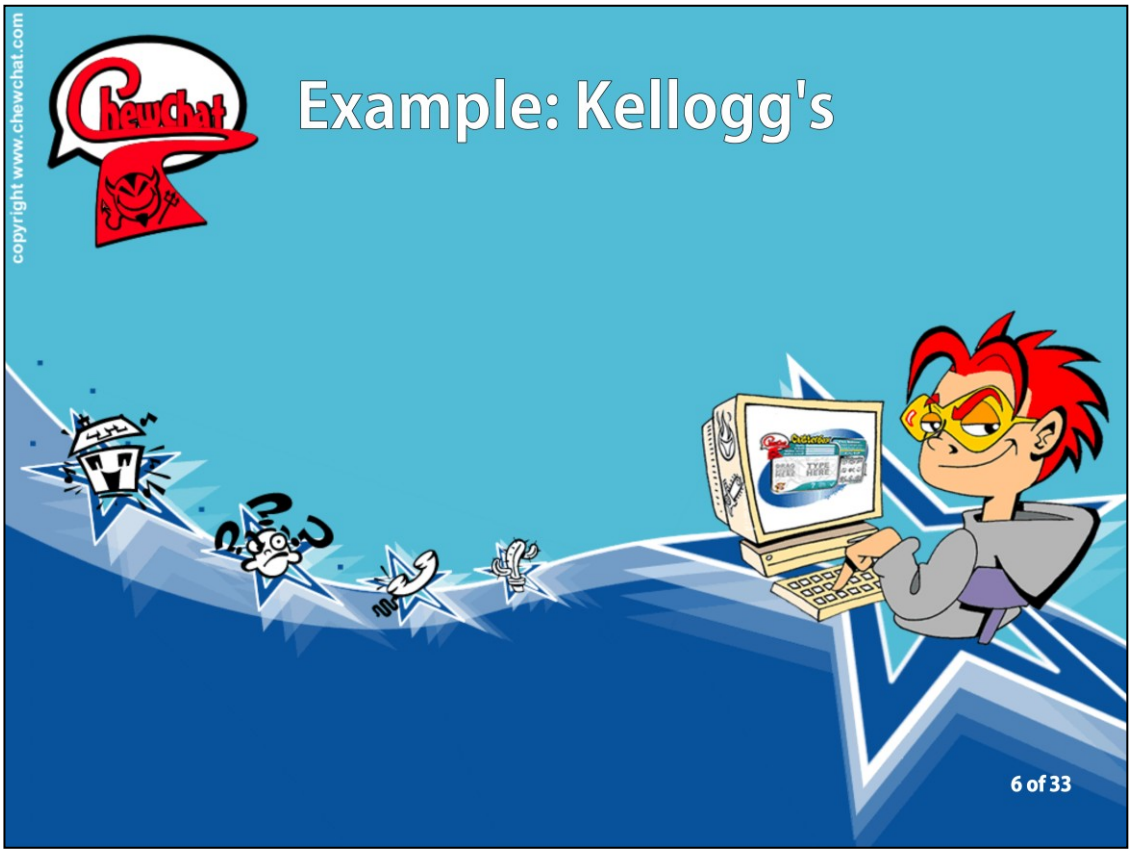
World-class clients: Kellogg's, Morgan Stanley, McDonald's, Heinz.

Awards, including 1st Cannes Cyber-Lions .

Power, money & excess.

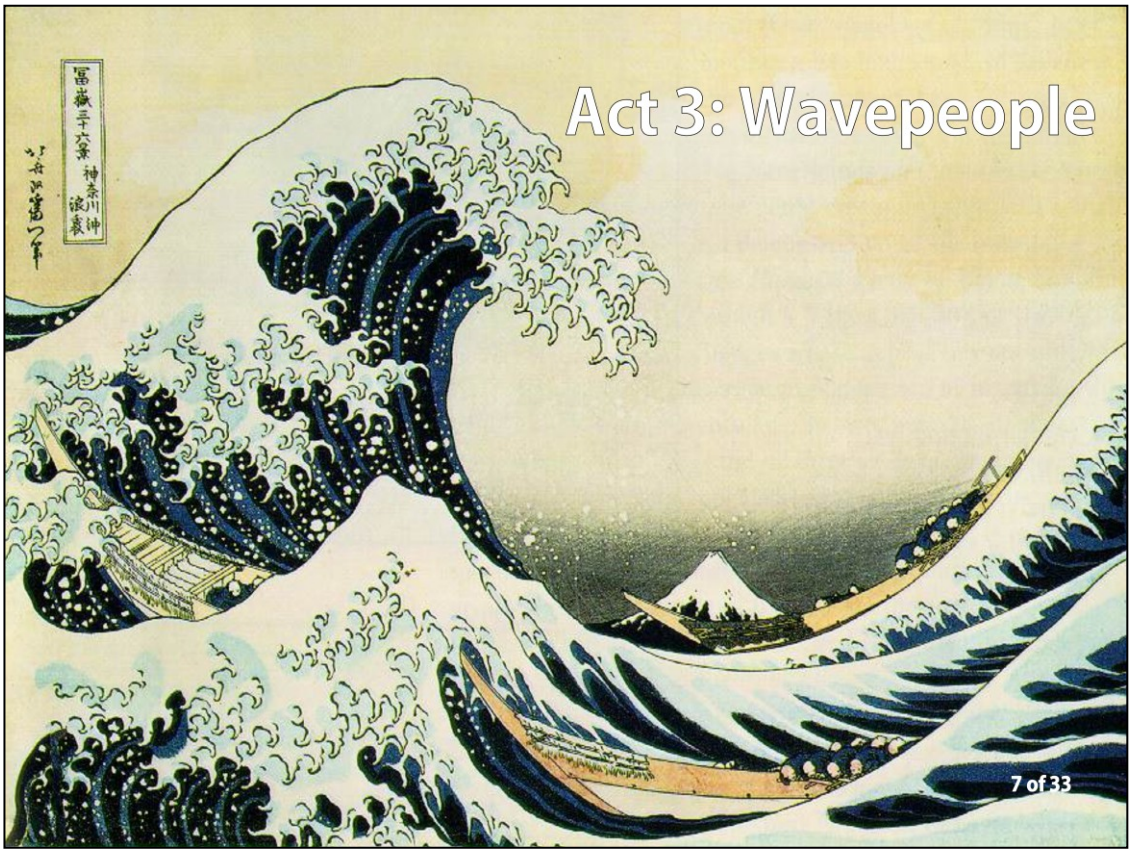
Money doesn't buy you happiness. Money doesn't even buy you money.

Management wasn't fun. I was lost.
Keep sight of where you are, where you've come from & WHERE YOU'RE GOING. Don't always go with the flow. Be brave.



(Steve Williams)

Case study in building a complete cross-media campaign from playing with the product.



2001: Going Ronin.

Spent “more time with my family” (not a euphemism).

Working from home.

Doing more ethical work – public sector/charity.

Too many roles: pitching, pricing, server admin, email admin, Internet admin, Health & Safety, managing freelancers

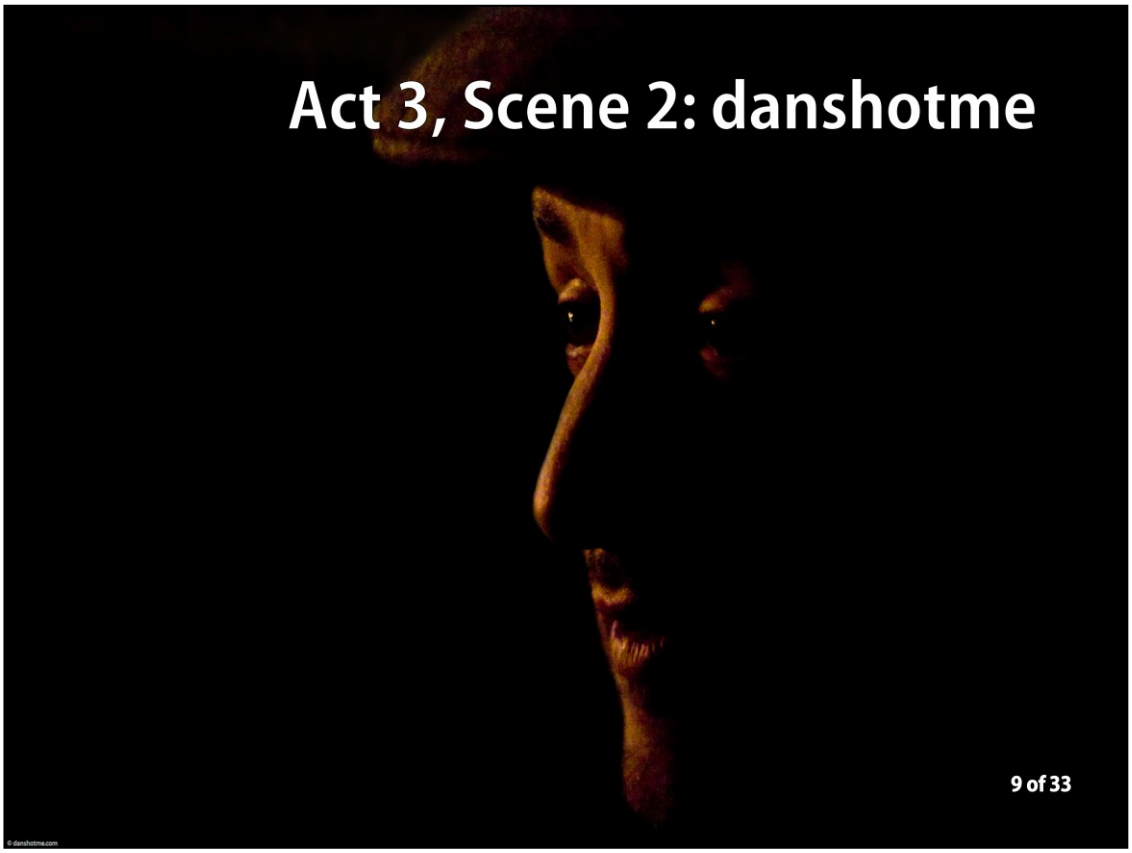
...and some coding

Learnt that freedom isn't everything. Work is social.

How to work with your target audience.

Engaging with kids, channeling their creativity & enthusiasm through our talents.

…and having a lot of fun.



Danshotme.com

Taking a sabbatical.

Social networking: MySpace -> Facebook -> Twitter

Cluetrain Manifesto.

Becoming a local celebrity.

Sidenote: FADwebsite.com

Act 4: iCoder

2007. Back in the modern world.

Commuting again.

On the jobs market.

Discovering how behind-the-times my understanding of web technologies had become over 6 years.

Focusing on one speciality: ActionScript.

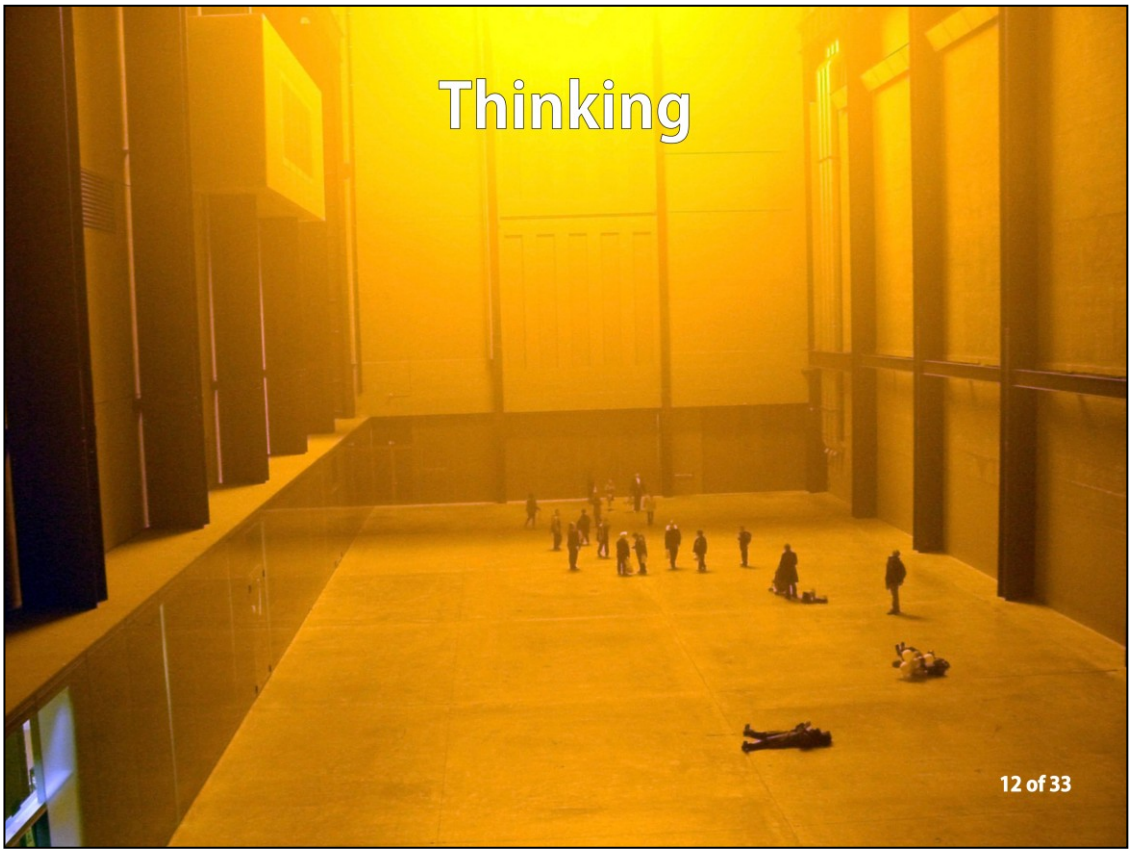
Learning new technologies & skills.

Getting to grips with methodologies & processes.

(No example: we've all seen iPlayer, right?)

Some useful stuff, and some obvious stuff.

The obvious stuff is more useful than the useful stuff.



Seems like most obvious thing in the world. It's not.

Easy to go on autopilot/cruise control.

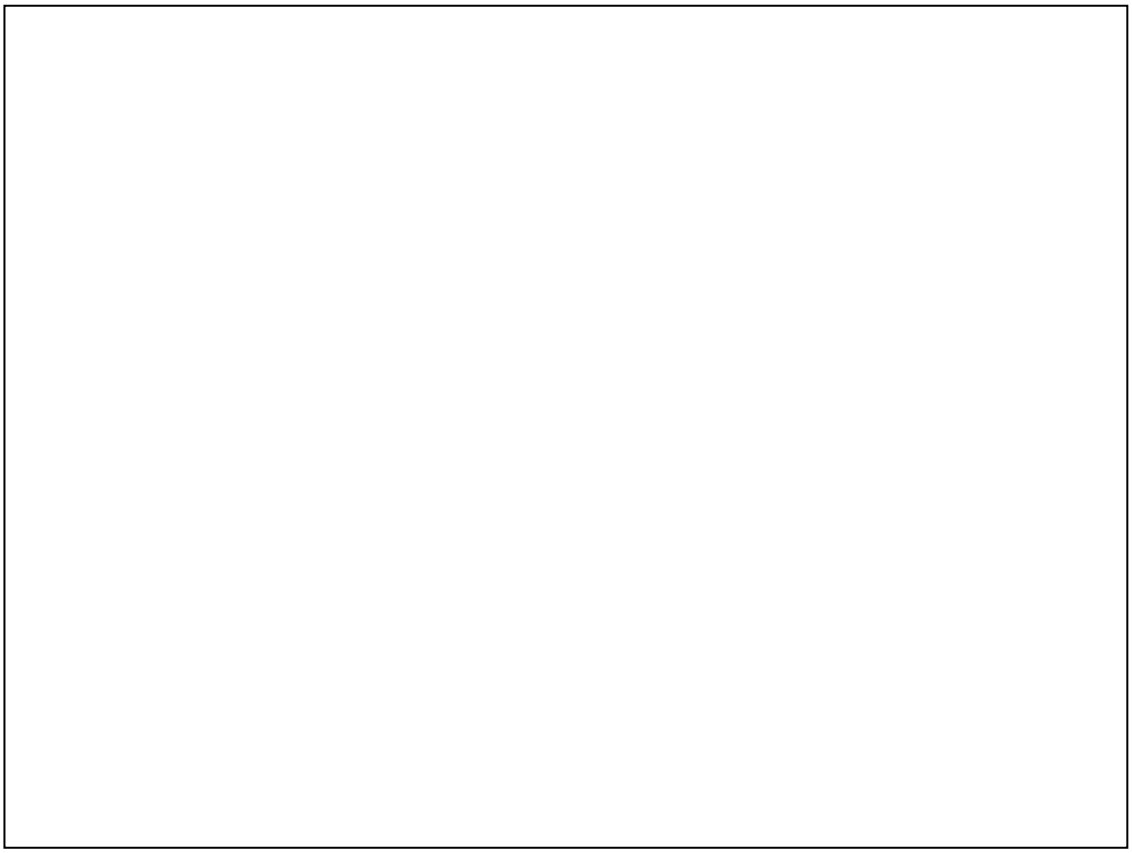
I've wasted whole years of my career without REALLY thinking.

Think BEFORE you start. Think DURING.

Buddhism: Mindfulness. Consciousness of being in the moment.

Meditation.

...is...



...a blank sheet of paper.

(Not lined; not squared; blank).

(And it helps if you have a Fisher Space Pen).

Why Don't You Just Turn Off Your Computer And Go Do Something Less Boring Instead?

Using a computer leads to using the wrong tools for the job (do I use Powerpoint? Viso? Illustrator...), forcing it to adapt to how we think.

Risk of distractions: regularly checking email lowers effective IQ by 10 points.

Computers kill productivity.

Walk for inspiration.

Bruce Chatwin – Songlines – Humans are nomadic walkers.

Constant low-level stimulation frees your mind.

Don't isolate yourself.

Rubber duck.

2 heads are better than one.

Pair code.

Peer review.

Share your concerns – be open in all things.

Secrecy is a recipe for disaster (somewhere down the line).

(Seb Lee-Delisle)

Read. Write. Understand.

Embrace everything.

You can be anything you want to.

You are not an ActionScript developer.

CLARITY before you write the code.

Know what it is you're writing.

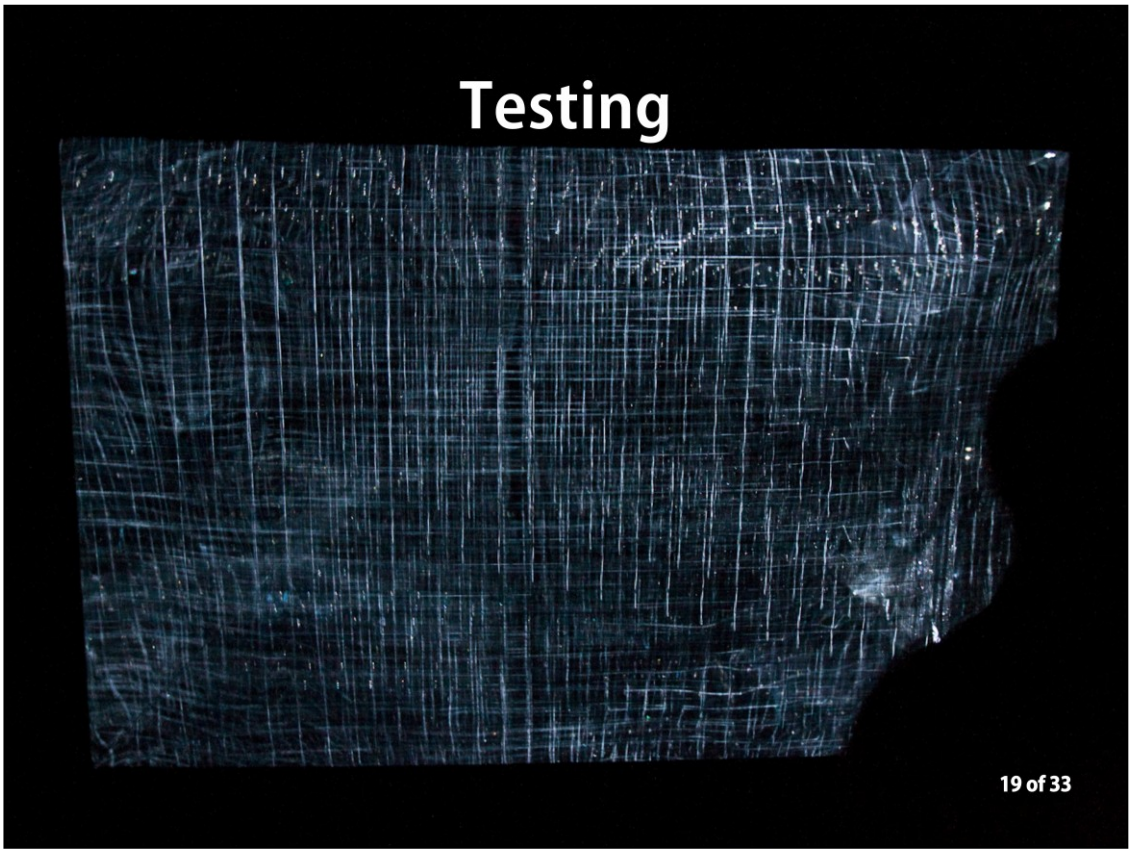
DON'T WRITE CODE YOU DON'T UNDERSTAND.

Don't fall into the trap of saying “oh well. That seems to have fixed it” and then moving on.

Use OOP & Design Patterns – take advantage of others' experience.

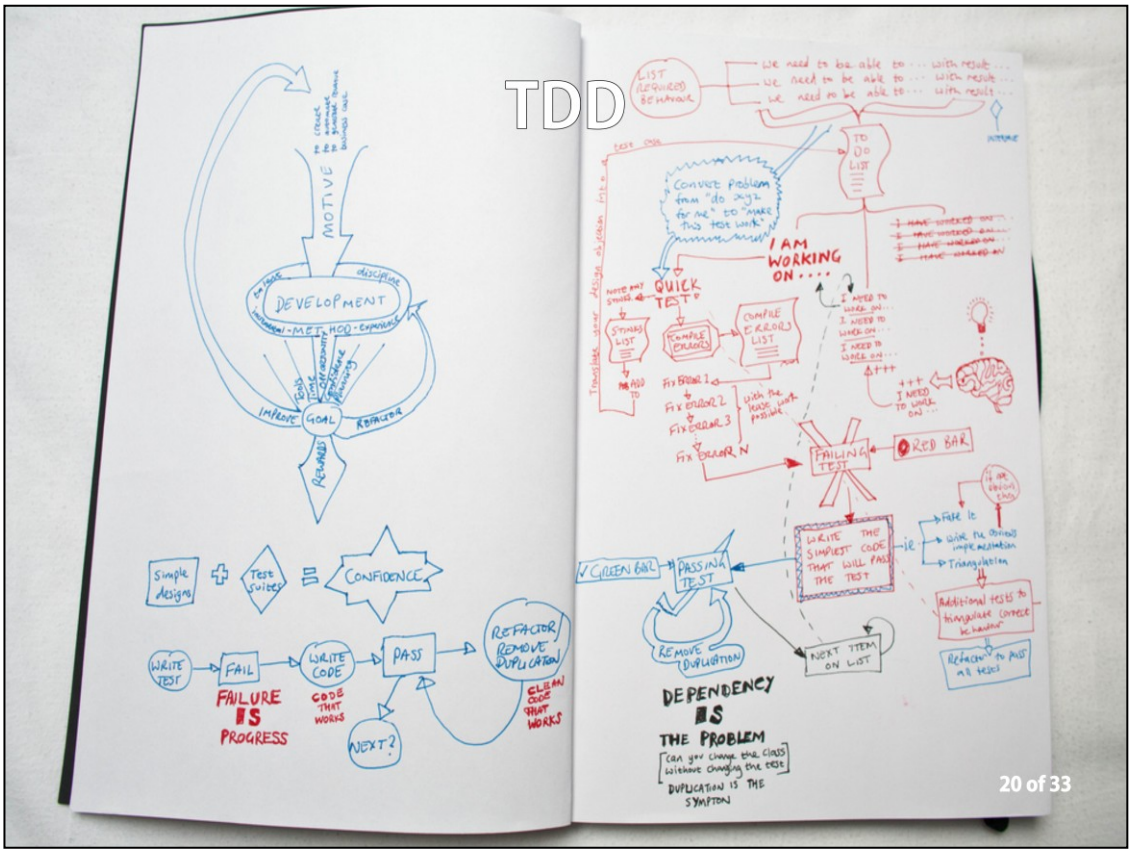
It's sometimes easier than you think to write from scratch...

...and harder than you think to adapt an existing code-base.



Testing can seem scary/time-consuming/pointless.

No system is complete without tests.



Gives you CONFIDENCE to change the code

Gives you CONFIDENCE to know what code you need to write

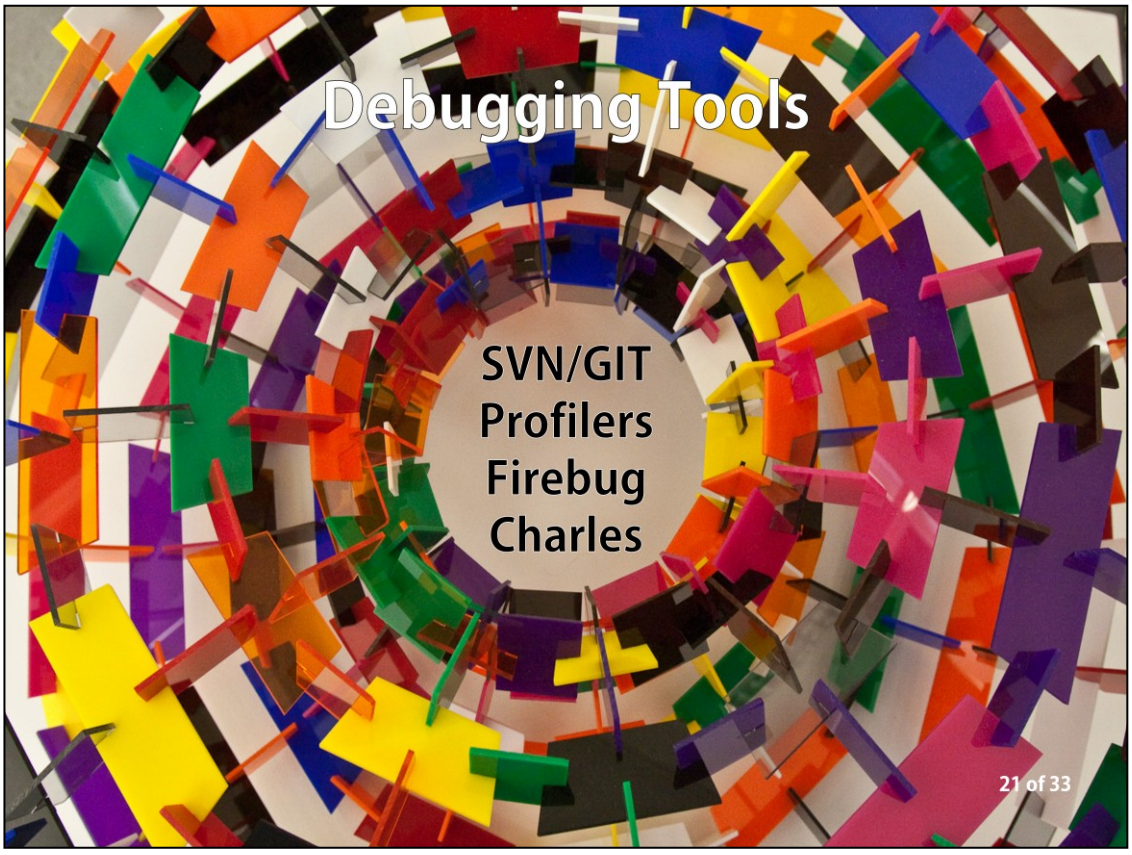
Lets you stick to taking baby steps.

Learning to be lazy – don't make me think.

What to test: Interfaces.

Use Mocking.

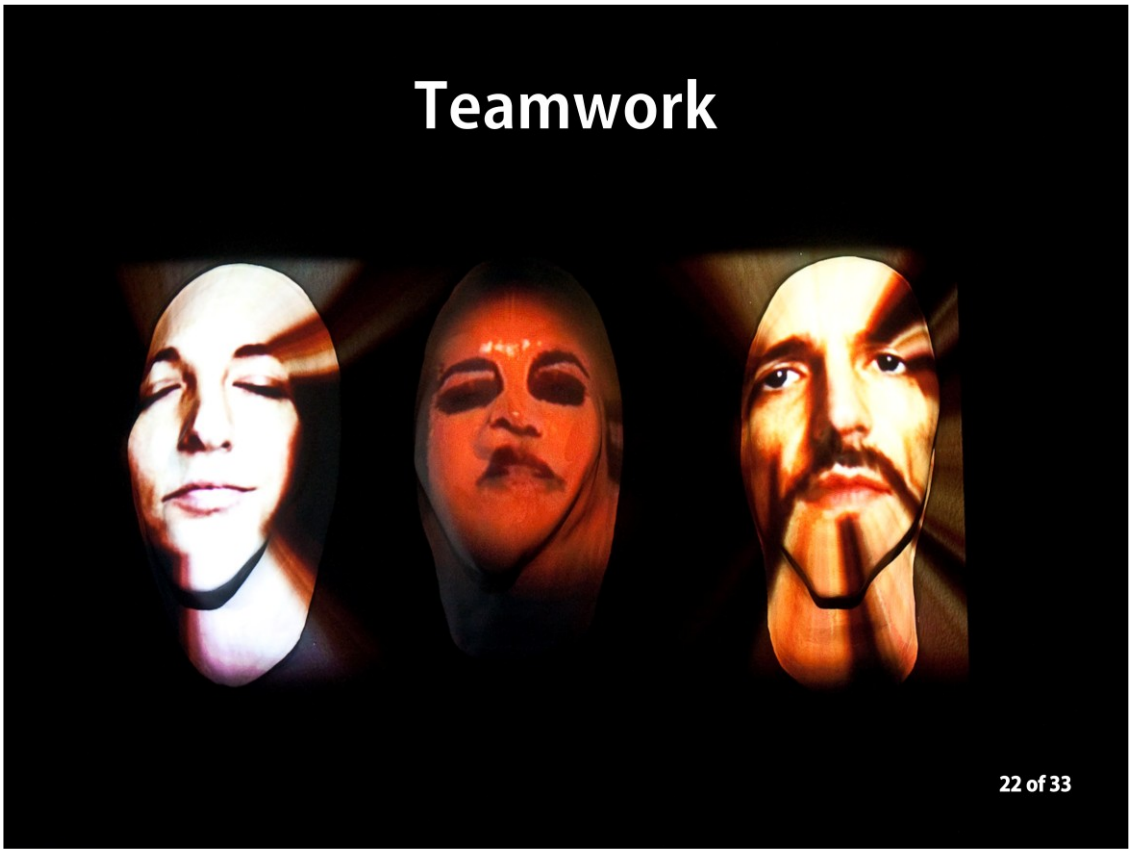
Plan / Test (red) / Develop (green) / Refactor (green).



Don't struggle on in ignorance

Get to the source of the problem -

BY ANY MEANS NECESSARY



Most projects require more than one person.

Remote working doesn't work (with caveats).

You need a process - “Having no process will kill you”

Administering the process is the responsibility of your Project Manager – but it is everyone's duty to make it work & continually improve it.

The process you choose depends upon your team – their relationships – and your work. There is NO one size fits all.

Start small – people hate change. Advance in achievable steps.

(Chris Pelsor)

(But “don't hire assholes”- Josh Hirsch)

Make an effort to know the people you work with.

Take an interest in their hobbies & passions outside work.

Adapt to the people you work with – change yourself, not others.

We see other people as constants, but their lives are as complex and variable as ours. Everything that goes on outside the office has an effect in the office.

Visibility of work

Catch problems early

Focus and K.I.S.S:

- What I worked on
- What I'm about to work on
- WHAT PROBLEMS DO I FACE

If one person's piece lasts more than (1 minute? 30 seconds?) – CALL ANOTHER MEETING.

Respect. Openness. Courage.

Ask the dumb question.

EVERYTHING should go on Wiki – to be searchable.

IMPORTANT stuff must be Talk/Email/IMed about ASAP.

Getting the balance right is hard, but it doesn't need to be perfect – every little bit of knowledge sharing helps.

No process fits every team
No process fits all the time
Track what works, and where the problems are, and fix them FAST
Spend time to save time – don’t let the urgent drive out the important.

Cards & avatars.

Visual tracking of work in progress.

Avoid task-switching: one job per person at one time.

Being agile – only do the minimum required for your task.

Don't be afraid of working with people who are better than you.

Don't be afraid to HIRE people who are better than you.

Find a bigger pond and you will inevitably start out as a smaller fish.

The individual as brand - super-freelancer.

It's in your interest for the product you work on to SUCCEED.

DELIVERING what you promise builds reputation

Sometimes it pays to say “no” - SPEAK TRUTH TO POWER

“Don't kid yourself, upper management WILL fuck you over given the chance”.

Have a game plan:

What am I doing/What do I want/What matter to me?

Know when to jump ship.

(Hoss Gifford)

Own up to them.

Learn by them.

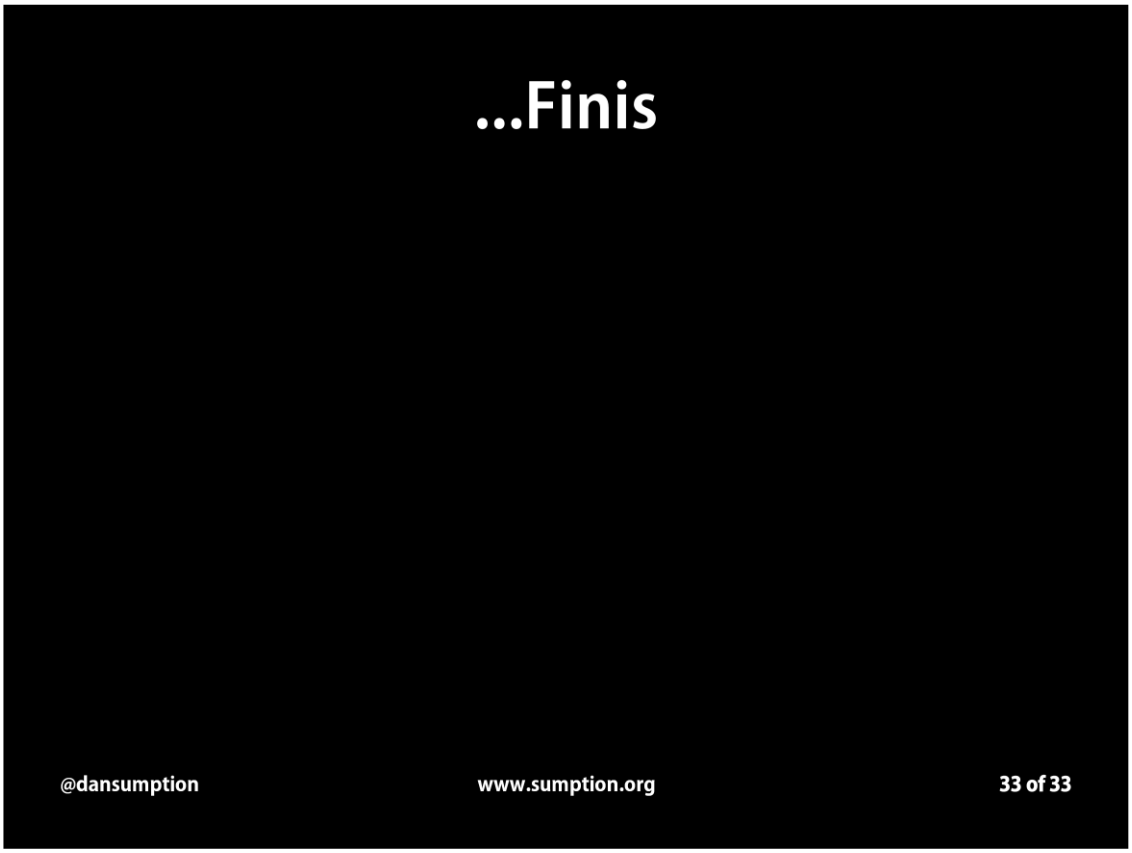
Learn & move on.

Fail more! Fail better!

"Clean Code: A Handbook of Agile Software Craftsmanship" - Robert C. Martin http://amzn.to/fWwqpM

"Pragmatic Thinking and Learning: Refactor Your Wetware" - Andy Hunt http://amzn.to/eQjreu

"Head First Design Patterns" - Eric T Freeman & Elisabeth R Freeman http://amzn.to/foJ0Bw



Thank you.