Stepping up your game

how an **investor mindset**
allows you to **improve your stack**

Arno Fleming

Tech Lead

The Next Closet

co-organiser

Amsterdam Ruby
Ruby NL

Hey, cheater!

...you're not an investor...

Still, this is

potentially

an interesting story

THE NEXT CLOSET

Before we talk about details

Let's get to know our context

The Next Closet

- 7 years old
- 2 founders, still active in the company
- emphasis on sustainability

- operate in the 2nd most polluting industry of this planet: fashion

- our mission: **second hand as the first choice**
- our niche: **designer fashion**
- our market: **Belgium, the Netherlands**
- total team size: **20 full time equivalents**

Welcome
on our journey
from start up to scale up

disclaimer

The talking points in this presentation are (hopefully) valid for The Next Closet. They may or may not apply to your business. When putting ideas from this presentation in action, you may come out better. Or worse.

Anyway, it's your business, and your business is not my business.

(I told you I wasn't an investor)

How an

investor mindset

allows you to

improve your stack

What is that

Investor mindset?

First and foremost:

Opportunity cost.

Does this investment
of my non-finite amount of money
bring me the highest benefit?

Opportunity cost

- **Assessing risk**

- **Assessing potential**

Investor: "There are no skeletons in this closet."

Opportunity cost

- Assessing risk

- Assessing potential

Investor: "There are no skeletons in this closet. Maybe in...

Opportunity cost

- Assessing risk

- Assessing potential

Do the current and proposed setup scale?

Investor mindset

That is not about business as usual

How an

investor mindset

allows you to

improve your stack

stack == stack, rite?

- ☐ Tech setup
  - ☐ security
  - ☐ IP
  - ☐ development processes
  - ☐ infrastructure

stack !== stack

- ☐ Tech setup
    - ☐ security
    - ☐ IP
    - ☐ development processes
    - ☐ infrastructure
- ☐ Continuity
- ☐ Team composition
- ☐ Culture

Tech::Security

The GDPR edition

- Will I be ashamed of all the data on the streets if they misplace their database?

- Who has access to the production database?

- Are customer credentials showing up in their (and possibly their vendors 😬) logs?

- Are personally identifiable information, on all disks, on everywhere machine that stores it, encrypted?

1. Gauge how you're doing:

https://gdpr.eu/checklist

2. Record your action points

Tech::Security

Cracker prep

- Have you pen-tested your app?

- Do you do static analysis of security violations in CI?
  (do you even do CI?)

- Do you employ bug bounties?

Record your action points

1. Zerocopter? HackerOne?
   💡 Ask your community for help

2. brakeman
   bundle-audit
   dawnscanner

3. responsible.disclosure@yourdomain.org

Tech::Security

Unwanted access

- Who has shell access?

- How and when are keys, tokens rotated?

- How are credentials shared?

- How do you handle brute forcing / intrusions?

Record your action points:

1. One shell user per human

2. On/off-boarding docs for credential rotation

3. Have some end-to-end encryption setup within your team

4. gem install rack-attack
   gem install prorate
   apt install fail2ban
   (RTFM)

Tech::Security

After it has gone wrong

- How often do you iterate on your backup & restore processes?

- Do you have an incident log?

- Do you learn from your incidents?

Record your action points:

1. Backup and restores are highly variant. Exercise it. multiple times per year.

2. Log your incidents in a spreadsheet, Notion, Asana, heck even Confluence

3. Write a 'Reason For Outage' doc, and present it to everyone (internally) who's interested.

Tech::Security

Code hygiene

- Which systems access your code?

- What parts of the code have compartmentalised access?

- Are credentials to external systems in the ENV, not the codebase?

Record your action points:

1. GitHub/GitLab?
   Developer laptops?
   Staging / Production servers?
   CI?
   Alerting / monitoring software?
   Forks of your repo 😬?

2. Build teams, give access based on roles

3. Apply tips on 12factor.net

Tech::Security

tl;dr:

Usually ROI means:

Return On Investment

In security, ROI usually means

Risk Of Incarceration

- ☐ Tech setup
  - ☑ Security
  - ☐ IP
  - ☐ Development processes
  - ☐ Infrastructure
- ☐ Continuity
- ☐ Team composition
- ☐ Culture

Tech::IP

(legal) Ownership

- Do we own all domain names we communicate?

- Do we own the code, and all ideas that form while at work?

Record your action points:

- Talk to your lawyer

- Make sure employees sign off their rights, and make sure they're compensated.

- Again: talk to your lawyer

Tech::IntellectualProperty

tl;dr:

Talk to your lawyer

- ☐ Tech setup
  - ☑ Security
  - ☑ IP
  - ☐ Development processes
  - ☐ Infrastructure
- ☐ Continuity
- ☐ Team composition
- ☐ Culture

Tech::DevProcesses

Ideation → Implementation

- What project management methodology do you use?

- Does tech debt grow or shrink?

- Does the backlog have a sane size?

- How often do you ship (even on Friday?)

Record your action points:

- Order your tech debt, from dumpster fire to WONTFIX

- (Re)assess your process, before requests land in your to-do board.

- (Re)assess your team composition: Do the engineers work with the right stakeholders?

- Increase the observability of your platform to instil trust in both the team and your platform

Tech::DevProcesses

How tech is used

- Are the programming languages, libraries and their dependencies current?

- What's the code to test ratio, what's the coverage for your primary processes(*)?

* - primary process: This is how you resolve the tension of your customers.

Record your action points:

- Use depfu / dependabot /other

- Follow (security) mailing lists

- Figure out (and plan to increase) the coverage for flows that matter most, and the units involved.

Tech::DevProcesses

How tech is created

- Do you XP?

- or Pair programming?

- or just PR reviews?

- Not even that?

Record your action points:

- Create a process to share knowledge, make the creation of software, not plenary meetings, elemental in that process.

- Make sure to protect the main branch, make reviews mandatory (but pair programming trumps a review)

Tech::DevProcesses

How tech is communicated

- Do you have release notes? Of everything? Or only end-user impacting parts?

- How current is your documentation?

Record your action points:

- If your feature was worth building, it is worth demoing, and it is worth writing release notes for. Start writing it.

- Ensure your documentation is connection to your code. Make sure it cannot lie. Optimise for the next hire, they _will_ join.

- Document code with test. Document concepts inside and outside your codebase.

Tech::DevelopmentProcesses

tl;dr:

Others (may) decide strategy, but each team makes all operational decisions.
The team decides on their tactics.

- ☐ Tech setup
  - ☑ Security
  - ☑ IP
  - ☑ Development processes
  - ☐ Infrastructure
- ☐ Continuity
- ☐ Team composition
- ☐ Culture

Tech::Infra

Exceptional cases

- Do you track client and server errors?

- Do you keep track of outages?

- What is your escalation procedure?

- What's your uptime? And more important: What's your definition of up?

Record your action points:

- Use AppSignal / Sentry / Honeycomb

- Share and present RFO documentation

- Make sure roles, purpose and accountabilities of engineering support are clear, make sure everyone knows who is/are one level up (to the CEO)

- Go beyond a working homepage for uptime monitoring.

Tech::Infra

Does it scale?

- Do you load-test your app?

- What are your performance bottlenecks, how did they evolve?

- How is your server health?

Record your action points:

- Start load testing _all the things_.

- Know where the pain points are, quantify how bad they are.

- Do you have over-capacity? Are you over capacity? What breaks if you are in the news?

Tech::Infra

When it hits the fan

- How is your backup / restore procedure?

- How do you host?

Record your action points:

- Reiterate your backups and restores. At least quarterly

- Make sure you are ready to handle OVH kind of issues. -> Learn from others' demise

Tech::Infra

DevOps

- How far are the differences between dev/qa/staging/production?

- How do you deploy?

- Do you auto-scale?

Record your action points:

- Use as few environments as possible -> No stress if staging is down, if it doesn't exist to begin with.

- Use feature flagging in production to get rid of everything between dev and production

- Is capistrano enough?
  Heroku?
  Terraform and Kubernetes?

  Pick whatever works with as little painkillers as possible

Tech::Infrastructure

tl;dr:

Never let a good crisis go to waste

- Winston Churchill

- ☑ Tech setup
  - ☑ Security
  - ☑ IP
  - ☑ Development processes
  - ☑ Infrastructure
- ☐ Continuity
- ☐ Team composition
- ☐ Culture

This was all ~skeleton searching~ risk assessment

Let's talk about ~non-boring things~ assessment of your stack's potential

The teams...

- ☐ continuity
- ☐ composition
- ☐ culture
- ☐

Let's talk about ~non-boring things~ assessment of your stack's potential

The teams...

- ☐ continuity
- ☐ composition
- ☐ culture
- ☐ 10X engineers

Not only 10x engineers
can add servers
to handle 10x load

Not only 10x engineers can build infrastructure no one understands

Not only 10x engineers can secure a website from intruders

Not only 10x engineers
can write code
no one wants to own

Your bottom-up culture

will help you grow

Your bottom-up culture

will help you grow

make

team members responsible

for finding and recording the

purpose and accountabilities

of the work they perform

Your bottom-up culture

will help you grow

connect the

purpose and accountabilities, or

roles

visible for the whole organisation

to the team members

that engage in these actions

Your bottom-up culture

will help you grow

Iterate on the roles

at least on a bi-weekly basis

and ensure

new joiners

follow suit

Examples, please!

- A team member that wants to join hiring?
  ➡️ Let them make a role (purpose + accountability) for that, while doing it.
  And hold them to it.

- Does anyone notice a mismatch between role and reality?
  ➡️ Have the role owner(s) change the role, to better match the reality.

Organically grow your culture

☑ roles, purpose and accountability

Organically grow your culture

- ☑ roles, purpose and accountability
- ☐ hiring & training

Hiring
&
Training

Use the roles

that do not receive enough energy

to drive your hiring,

and use roles that are unclear to

drive your teams' education

Hiring & Training

Interviewer bias?

Use a standardised set of questions
in your hiring process.
Otherwise you may end up
with a standardised set of colleagues.

Hiring & Training

What happens if we train them, and they leave?

- some boardperson

Hiring
&
Training

What happens if we don't train
them, and they stay?

- their CEO

Organically grow your culture

- ☑ roles, purpose and accountability
- ☑ hiring & training
- ☐ action

Action?

Action

is something that can be done with

Agility

And these concepts may shadow each other when talking about work.

But should they?

Action?

When in doubt,
always take action

- an engineer in a slow
deciding organisation

Action does not scale

it doesn't sound as seductive as 'Action', but

Processes do scale

If action == agile

Process == Lean

Think first, so you'll act faster

When in doubt, formulate a process

Action?

Wait. Let's make that more seductive.

Process over Actions

- Arno Fleming

Organically grow your culture

- ✅ roles, purpose and accountability
- ✅ hiring & training
- ✅ ~action~ tactics

underpinning all of your growth plans:

process over actions

1 Remove the skeletons from the closet

0 Create processes to scale your culture.
   Iterate on these.
1 Remove the skeletons from the closet

0 Create processes to scale your culture.
   Iterate on these.
1 Remove the skeletons from the closet
????
PROFIT

Fin

any questions?

(Also find me on Slack: netherlands.slack.com #amsterdam)