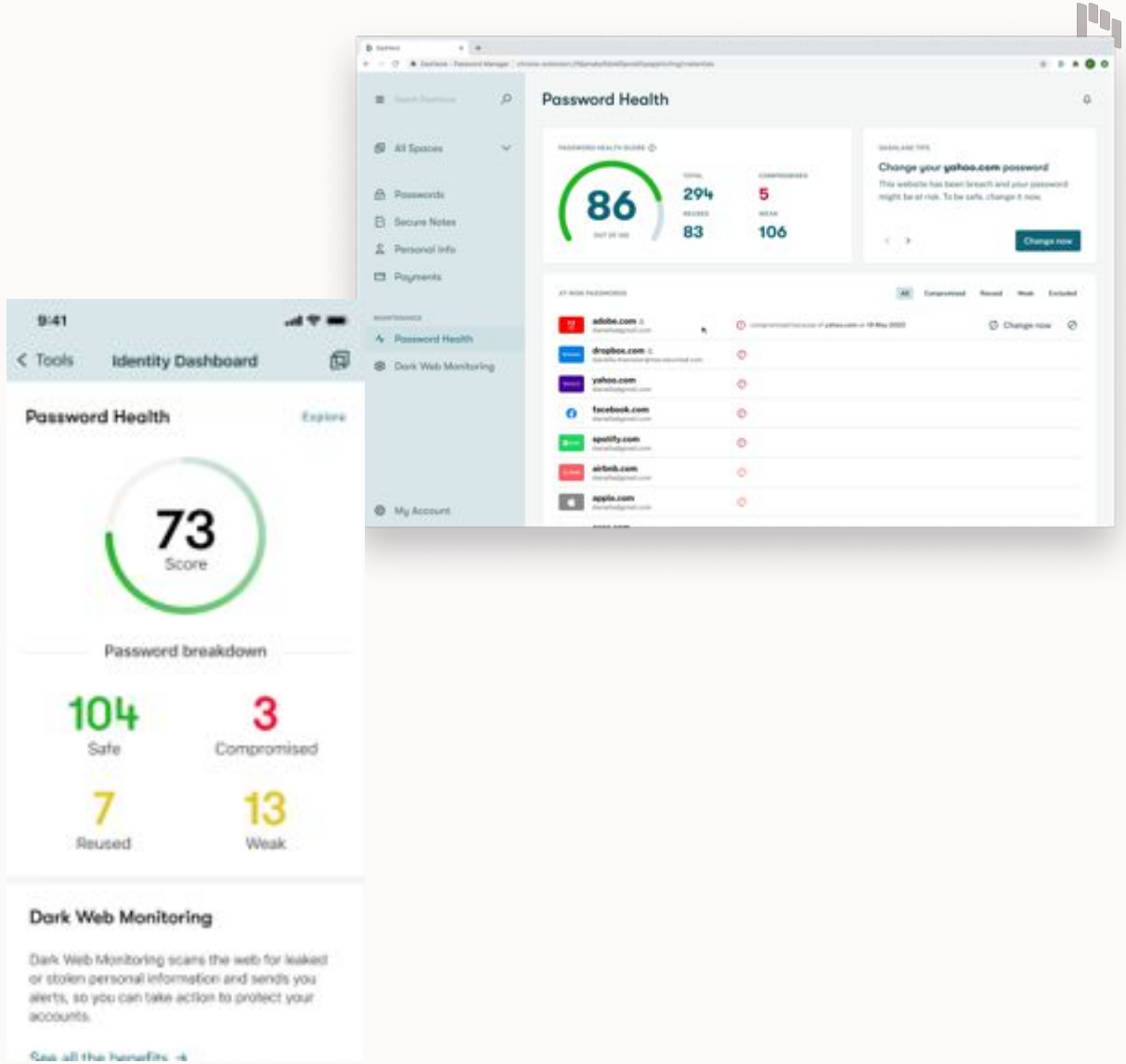
Continuous Learning

My name is Frédéric Rivain, CTO of Dashlane

We build a Password Manager, to help you manage your identity and your payments in a simple and secure way everywhere.



A bit of context

Founded in 2009 by *Bernard Liautaud* and 3 French students from Ecole Centrale Paris

~270 employees in Paris, Lisbon and New York

Consumer product (B2C) + Enterprise offer (B2B)

~15 “product & engineering” teams

100+ Engineering Team

Developing a Culture of Learning

Performance

- Teams that learn quickly are more adaptive than teams that don’t.
- Adaptive teams are teams that can get better results, by rapid response to change.

Happiness

- Fulfilling the need for progress and mastery
- Not being bored

A bit of theory to start with

Levels of learning

3 levels of interdependent learning:

1. Individual
2. Team
3. Organization

Continuous Learning at *INDIVIDUAL* Level

Learning requires time and effort, as well as the decision to want to learn.

Make individuals understand the value of continuous learning, and how it will not only help the organization, but most importantly, it will be a great benefit to the learner as well.

Examples:

- Trainings;
- coaching and mentoring;
- seminars and workshops;
- And mostly through actual on-the-job application and practice of skills and knowledge.

Continuous Learning at TEAM level

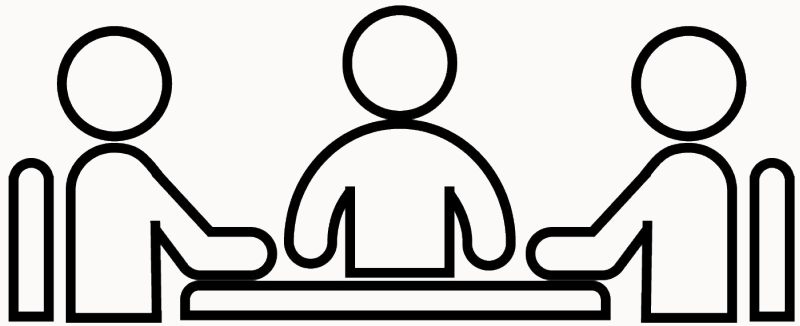
Means collective individual learning:

- if the members of the team acquire and share new knowledge and information, then team learning takes place.

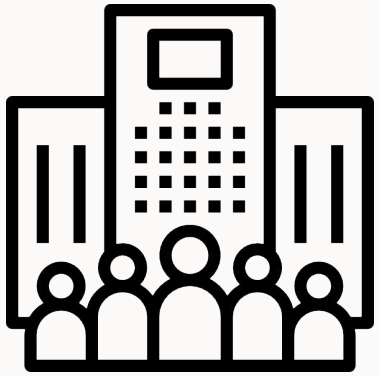
Involves a set of learning processes that support and aid team performance

Examples:

- team sharing or training,
- agile retrospective,
- feedback,
- experimentation,
- group discussions and Q&A sessions.



Continuous Learning at ORGANIZATION level



Comprises **change** of interaction patterns, change of policies and procedures, **new culture** and new innovations.

Examples:

- feedback from the employees themselves, from clients, and from customers.
- Getting comments and ideas.
- Culture and Change Management.
- Processes.

Keep in mind:

Tribal Learning

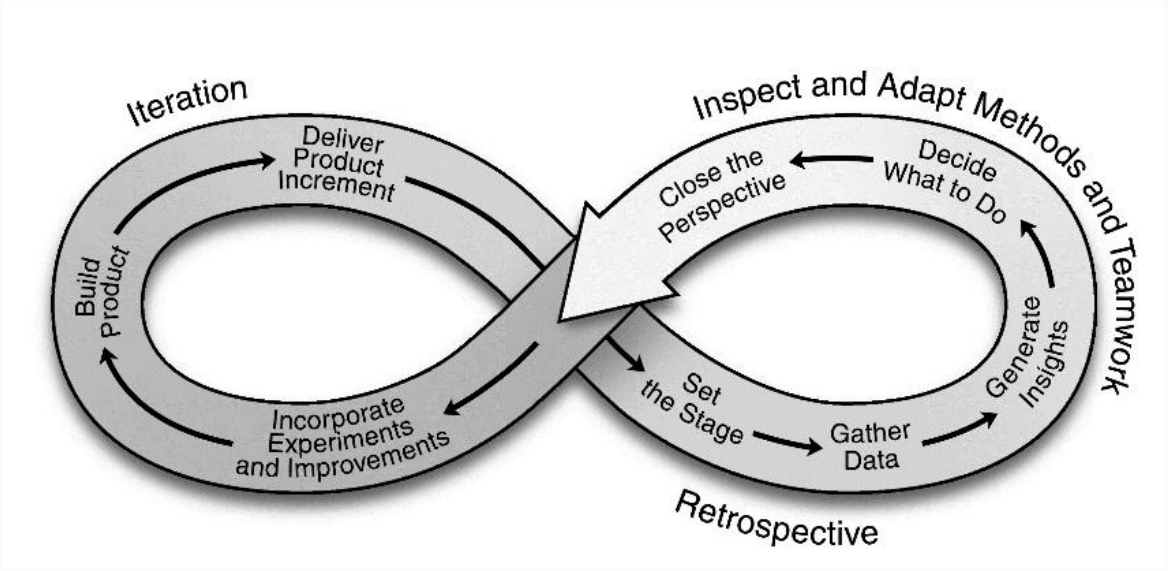
Dunbar's number: 150

Agile: a learning laboratory

Agile means that teams must first become skilled at learning as a group:

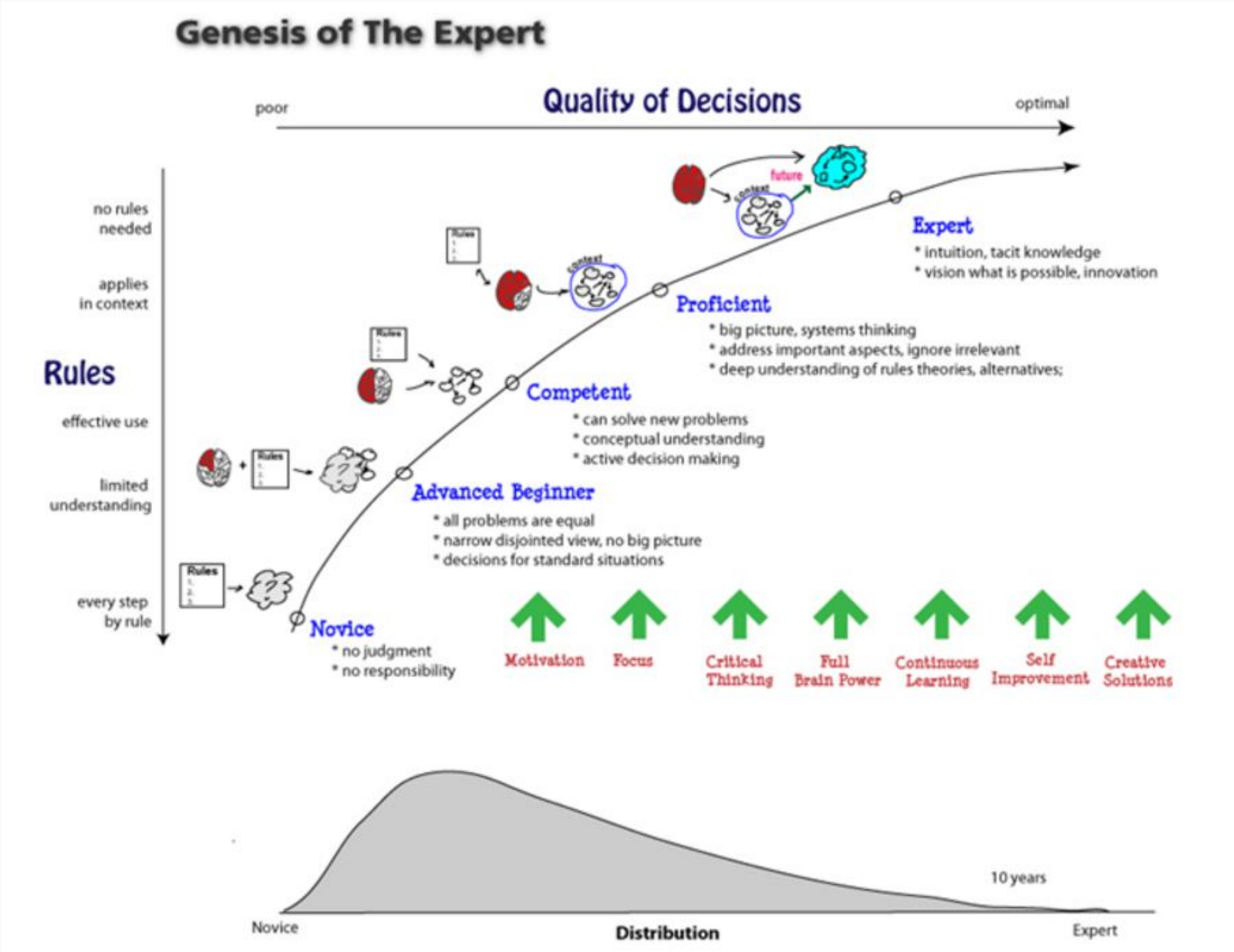
- Retrospective
- Auto-organization: estimate, design, self-management
- Safe space: safe to take risk. Experiment / trial-and-error: fail or succeed.

Agile teams are in fact small Learning Organizations.



Dreyfus Learning Model

https://en.wikipedia.org/wiki/Dreyfus_model_of_skill_acquisition



10 000h Model

10 000 hours of practice are required to master a discipline. Controversial model.

But practice is key to learning. Practice, practice, practice.

A professional poker player plays on average 100 000 hands a month / 17 hands per minute.

Shu-Ha-Ri

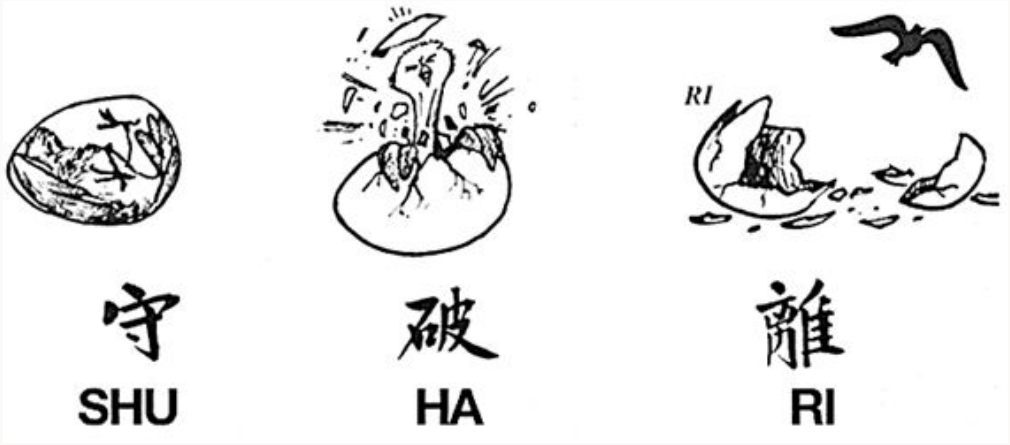
Introduced by Alistair Cockburn

Aikido reference

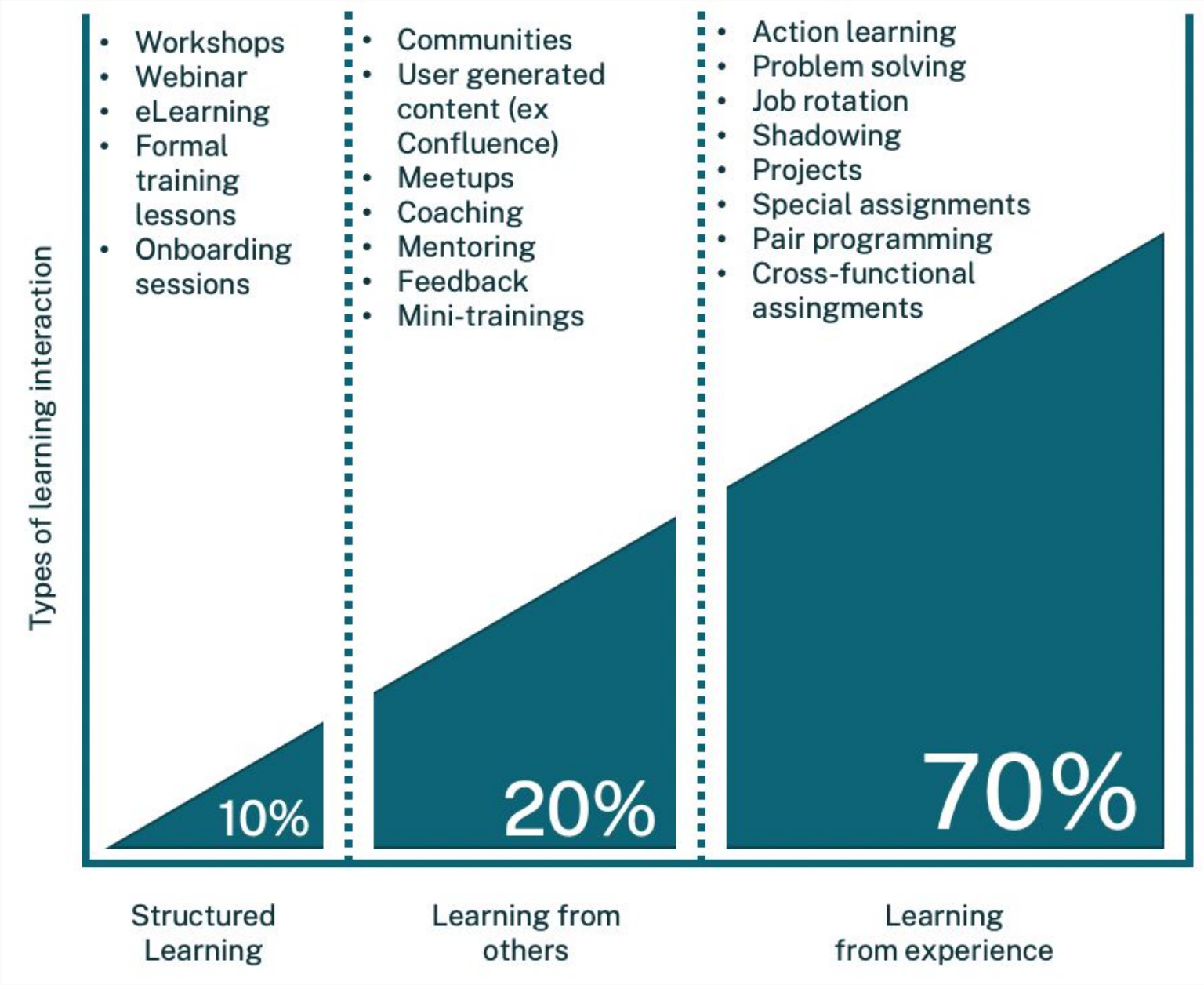
Shu : follow the rules (learn)

Ha : master the rules (become a master)

Ri : break the rules (create new rules, innovate, surpass the master)



70:20:10


Types of learning interaction
Workshops
Webinar
eLearning
Formal training lessons
Onboarding sessions
Communities
User generated content (ex Confluence)
Meetups
Coaching
Mentoring
Feedback
Mini-trainings
Action learning
Problem solving
Job rotation
Shadowing
Projects
Special assignments
Pair programming
Cross-functional assingments
10%
20%
70%
Structured Learning
Learning from others
Learning from experience

The Role of Tech Leaders

How to contribute to a culture of continuous learning

Be purposeful

Announce your intent

Learning requires purpose and intent

- State your purpose early and often. Make it easy for those who follow you to understand your vision, your mission and your intent.
- This **clarity** helps everyone around you and increases levels of group learning.
- Explain. Share your thought process.
- Again and again.

Teach each others

Conduct Frequent Experiments

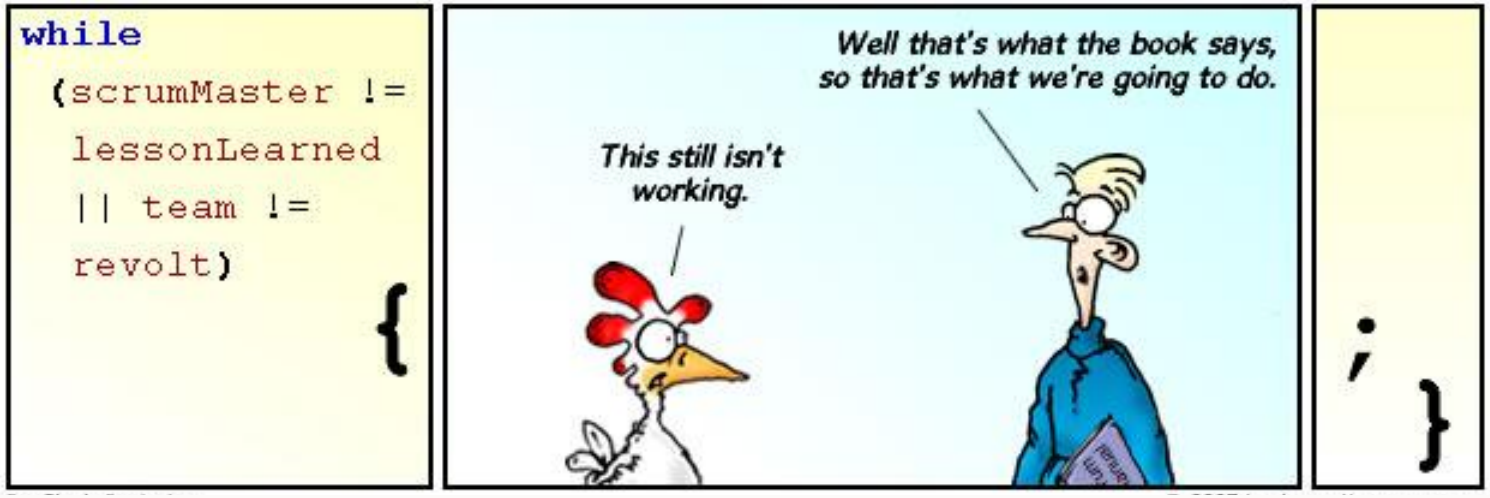
Frequent experimentation means frequent learning.

Make learning into a game, by scheduling frequent, cheap experiments.

Failing cheap means learning economically.

Inspect Frequently and challenge the status quo

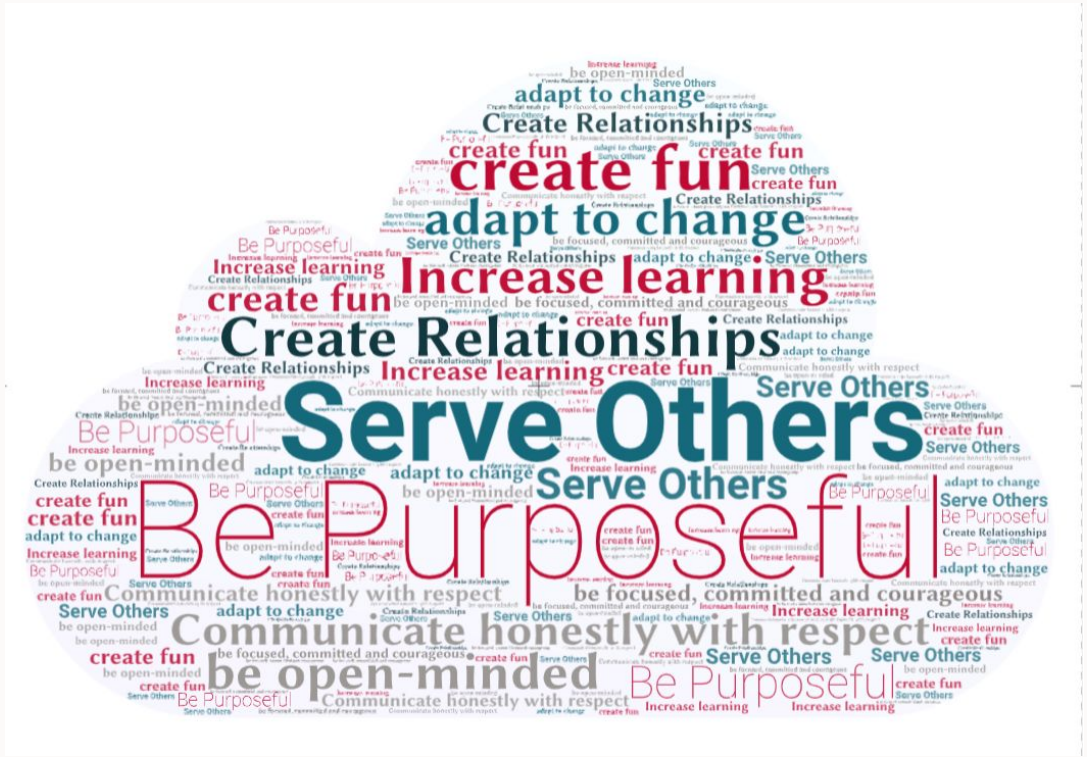
- Use iteration and frequent inspection to make a game of change. Inspect and retrospect frequently at all levels.
- Pay attention to what is working and what is not. Zoom in on details and focus on results. Discuss with the specific intent to be excellent.
- Retrospective at all levels.



Examine your values

- Normal is what you willingly tolerate. Examine your norms, because what you tolerate is a minimal level of what you insist on.

- Encourage good group behavior :
  - *paying explicit attention,*
  - *being punctual,*
  - *honoring values*



Understanding Delays

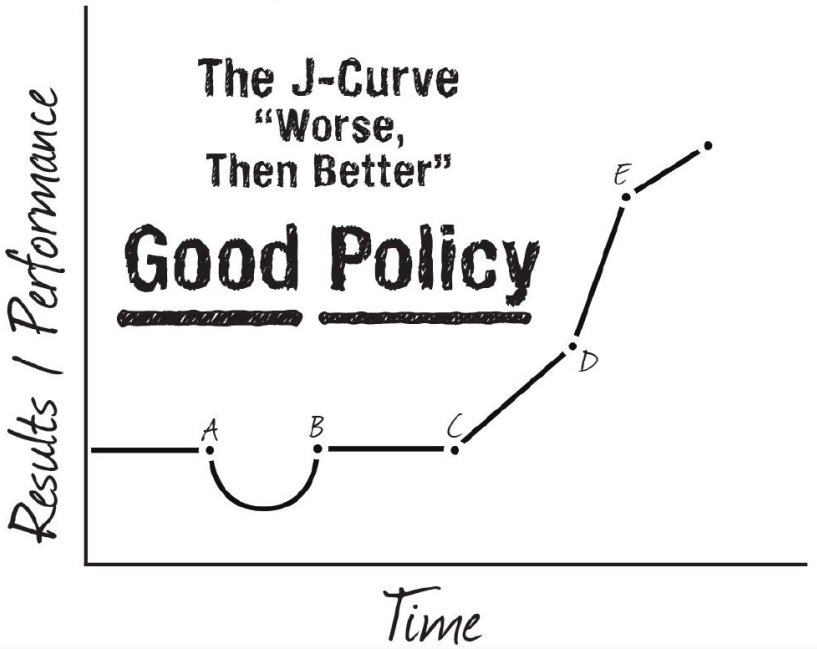
Delays in achieving good results are common.

Good steps taken today usually do not have an immediate positive effect. The truth is that **you often get worse before you get better**, because of the **investment period**.

Experiment cheaply.

Avoid the tendency to backslide to old habits, even if changing is painful.

Figure 12. The J-Curve



DASHLANE

Mini-training cycle

30 min weekly:

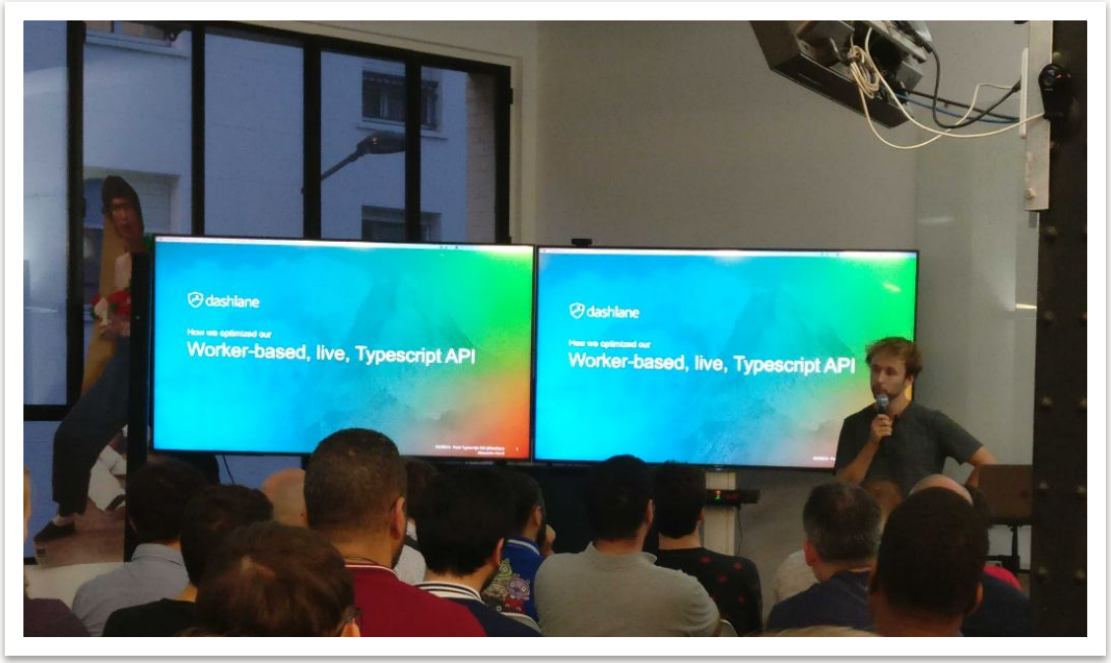
- 20 min presentation + 10 min question.

Any speaker, any topic.

Raise team awareness about continuous learning

Explore new topics with curiosity

Improve communication skills by being a speaker in a safe environment



Conferences & Meetups

Be a speaker or participate to conferences and meetups.

- Learn and discover
- Meet other people. Create relationships.
- Open your horizon
- Share your knowledge and experience
- Become thought leaders

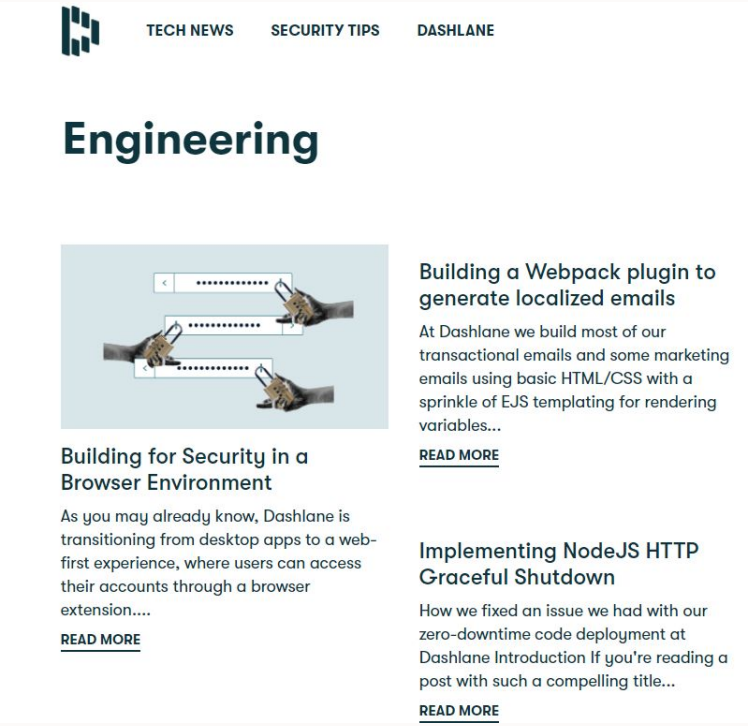
Blogs / Intranets...

Internal

- Share, Communicate, Serve Others.
- Increase Learning

External

- Promote your Engineering team, Motivate, hire, retain top engineers
- Contribute to team learning and performance, by reflecting, formalizing and sharing our practices (with the peer pressure of making it public)
- Attract external contributions



Brown-Bag Lunches

Invite an expert to come and speak to the team (and offer him lunch).

Another opportunity to learn, from the experts.

http://www.brownbaglunch.fr/

A variant: Lunch & Learn

- Invite outside people for lunch. Informally. No Agenda.
- Trigger the discussion with your guest to learn and explore new things.
- An informal version of the brown-bag lunch.

Hackathons / Innovation Days

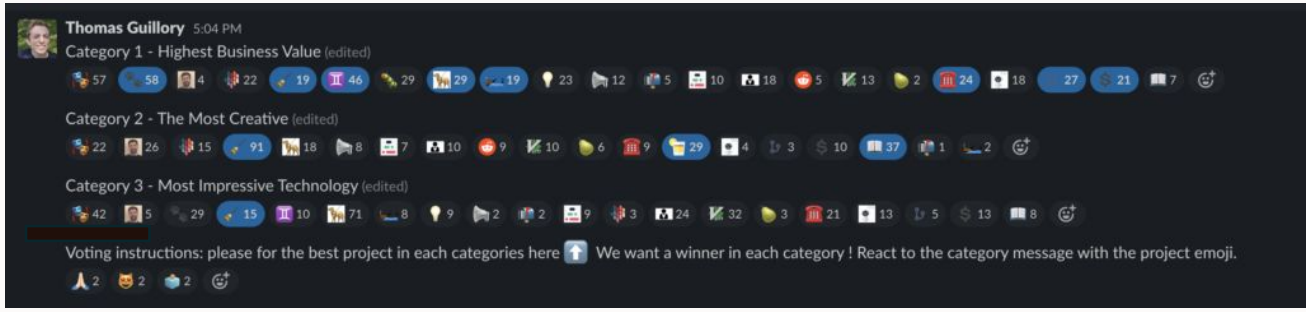
Take some time off to take a break and innovate:

- Build prototypes, demo them, vote for the best and reward the best team, follow up to put in production the best ones.

Be creative

Work with other developers and with business teams

Learn and have fun



Tech Weeks / Lab Days

Regularly organize days where developers are allowed to do other stuff (training, clean up code, prototype, open-source contributions, write blog articles...)

- Agenda for each person needs to be explicit.

Auto-learning

Developer Exchange Program

Switch developers between teams or companies to share and learn.

- Discover and learn other practices
- Be open-minded
- Create relationships

Pair and Mob Programming

Pair Programming is 2 developers working together: either for mentoring, or between peers on a complex topic.

Mob Programming: is an extension to a whole team to collectively train the team to a new technology or architecture.

- https://en.wikipedia.org/wiki/Mob_programming
- https://www.agilealliance.org/glossary/mob-programming
- https://mobprogramming.org/

Code Dojos & Coding Katas

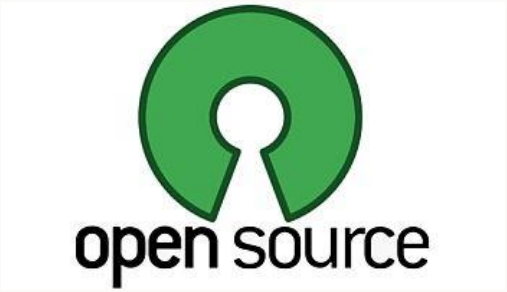
Play with Code: train your programming skills with small exercises, challenge your abilities and encourage to find multiple approaches.

Good Team Building moments

Plenty of resources online

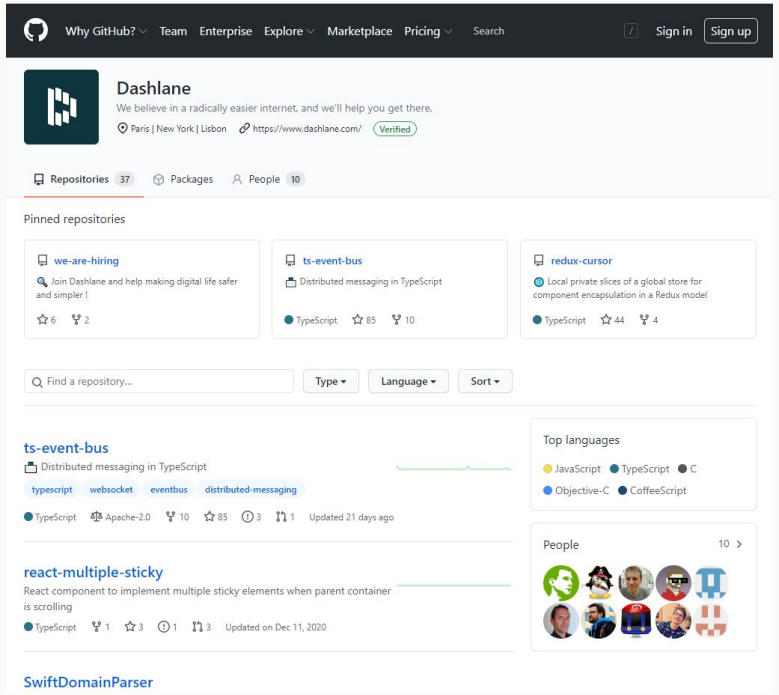
- https://www.codingame.com/
- http://www.codechef.com/
- https://codingdojo.org/
- http://programmingpraxis.com/
- http://projecteuler.net/
- https://codingcompetitions.withgoogle.com/codejam

External projects



Encourage developers to participate in external projects:

- https://github.com/explore
- http://sourceforge.net/blog/potm/
- http://www.codeplex.com/
- https://blog.dashlane.com/contributing-to-free-open-source-projects/



Open-Source your internal tools.

Learning with others

Practice other areas of coding

« Peer pressure » on code cleanup

Learning with fun

Best way to learn and bond

Team Building

Discover concepts in a different way

Few ideas

- Marshmallow Challenge
- Egg-Drop Challenge
- Lego4scrum
- Cocktail Factory
- Team Flags

DASHLANE

Final Thoughts

1. Be curious
2. Never take things for granted
3. Have fun

Thank you