CRAFT HEALTH CHECK

BY YOAN THIRION

WHO AM I ?

TECHNICAL AGILE COACH, SOFTWARE CRAFTSMAN

I'M YOAN THIRION (FREELANCE)

- DESIGN SOFTWARE SINCE MORE THAN 12 YEARS
- FUNDAMENTAL TO SUCCEED IN THAT AREA : AGILITY AND TECHNICAL EXCELLENCE
- HELP TEAMS DELIVER WELL CRAFTED SOFTWARE
- IMPLEMENTATION OF AGILE AND TECHNICAL PRACTICES (EXTREME PROGRAMMING, REFACTORING, DDD, MOB PROGRAMMING, ...)

MY SERVICES

TECHNICAL AGILE COACHING

COMMUNITIES OF PRACTICE

CULTURAL CHANGE

DEVOPS

BROWN BAGS

SERIOUS GAMES

LET'S CONNECT

FOLLOW ON

HTTPS://WWW.YOAN-THIRION.COM/

TRAINING / COACHING

AGILE DEV PRACTICES

- BOOTSTRAP TEAMS (SCRUM, KANBAN, XANPAN, ...)
- XP (TDD, PAIR PROGRAMMING)
- FACILITATION TECHNICS (EVENT STORMING, STORY MAPPING, ...)
- CODE KATAS
- CODE REVIEWS
- TESTING PRACTICES (PBT, APPROVAL TESTS, CDC, ARCHITECTURE TESTS, ...)
- MOB PROGRAMMING
- CLEAN CODE
- DDD
- RETROSPECTIVES
- CI/CD
- DESIGN SESSIONS
- CO-DESIGNS
- ARCHITECTURE REVIEW
- CONITNUOUS IMPROVEMENT / LEARNING
- ...

Learning hours

- Discover Practices
  - Pair Programming
  - Code Review
  - Interview Domain Experts
  - Dev Ethics
- Technical Debt retrospective
- Languages / Libs
  - F# for OO Programmers
- Testing practices
  - How to name our Unit Tests
  - Consumer Driven Contract Testing
  - Improve your test quality with Mutation testing
- Software Architecture
  - DDD re-distilled
  - NoSQL
  - Fundamentals of Software Architecture
- Agile Coaching
  - How to run a Community of Practices (COP)
  - The developers — the forgotten of agility
  - Xanpan
  - Drive and Intrinsic motivation
  - Leadership lessons from the Navy SEALs
    - Part 2
  - Agile HR
- Xtrem
  - En route vers l'apprenance avec Xtrem Reading
  - XTREM WATCH — Découvrez la puissance de la veille collective
- Good Morning Learning by Philippe Bourgau

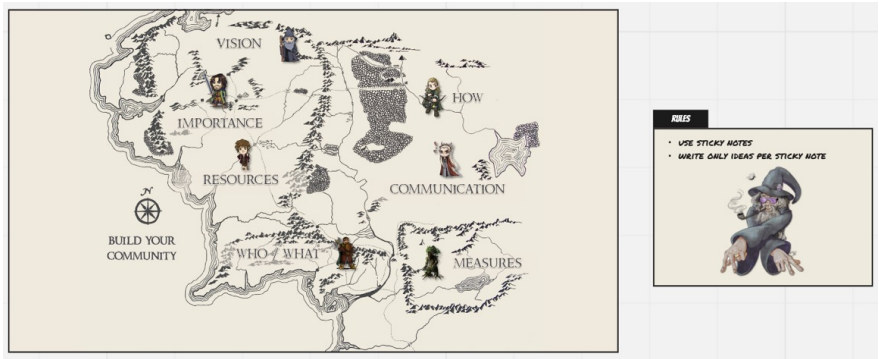
Code Katas

- Craft code
  - Clean Code
  - Write S.O.L.I.D code
  - Pure Functions
  - Clean Architecture
  - Functional Programming made easy in Java & C# (Vavr / Language-ext)
- Test Driven Development
  - Stack kata
  - Fizzbuzz kata
  - Outside-in TDD
- Testing Practices
  - Hands-on mocking
  - Property-Based testing
  - Approval Testing with Gilded Rose kata
- Refactoring
  - Mikado Method
  - Theatrical players refactoring Kata (OO and FP style)
  - Refactoring Journey
- Kata Catalog

Serious games

- Craftsminator : découvrez le Software Craftsmanship à l'aide d'un Escape Game à base de cartes
- Agile comfort zone
- Drive and intrinsic motivation
- Craft-challenges : discover Technical Agile Practices
- Yata - a DevOps game
- Customer Journey from mars
- DESPO'RATE HOUSEWIVES

https://yoan-thirion.gitbook.io/knowledge-base/samman-technical-coaching



COMMUNITY OF PRACTICES

- CONNECT PEOPLE / BREAK SILOS
- CONSOLIDATE THE COLLECTIVE KNOWLEDGE
- IMPROVE QUALITY
- SPREAD KNOWLEDGE
- STANDARDIZE PRACTICES IN THE ORGANIZATION
- SHARE PRACTICES
- BUILD NEW SKILLS / LEARN TOGETHER
- BUILD A LEARNING ORGANIZATION
- SOLVE PROBLEMS BY USING THE COLLECTIVE INTELLIGENCE

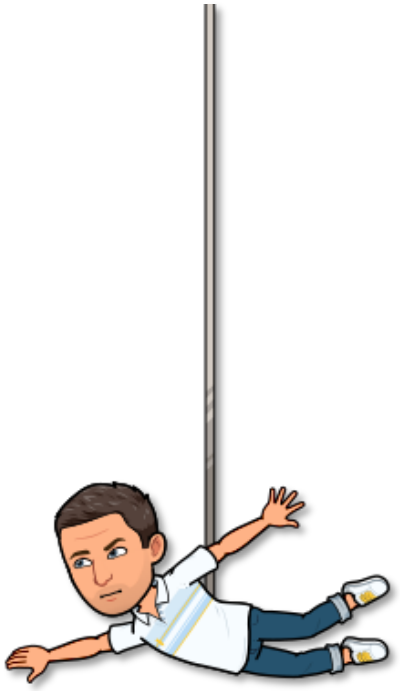
HELP THE TEAMS TO GROW TOGETHER / UPSKILLING WITH TECHNICAL, AGILE, PEOPLE SKILLS
MAKE THEM AUTONOMOUS ON THEIR PRACTICES & LEARNING

OBJECTIFS

FAIRE 1 ÉTAT DES LIEUX TECHNIQUE ENTRE NOUS AFIN D'IDENTIFIER DES ACTIONS D'AMÉLIORATION

COMME DANS 1 RÉTRO CLASSIQUE

- SOYEZ RÉALISTES
- OSEZ DIRE LES CHOSES
- PAS DE JUGEMENT
- PAS DE RECHERCHE DE COUPABLE
- RESPECT
- CE QUI SE DIT DURANT LA RÉTRO RESTE DANS LA RÉTRO

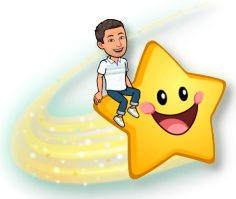


@YOT88

PLAN

- TOUR DE TABLE (10 MINUTES)
- CARTES D'AUTO-ÉVALUATION (1H)
- DISCUSSIONS LIBRES
- UN TOUR DANS LE CODE (20 MINUTES)

- PRIORISATION AXES / ACTIONS



TOUR DE TABLE

CHACUN(E) SE PRÉSENTE :

- NOM / PRÉNOM
- TON RÔLE / BACKGROUND
- CE QUE TU PRÉFÈRES DANS TON QUOTIDIEN

- 1 FAIT SUR TOI

"If I had to be trapped in a movie or a video game for a day, it would be…"

"An accomplishment I'm particularly proud of is…."

"One thing I cannot live without is…."

"One thing I know I do well (or better than most) is…."

"Right now, I'm reading about…."

"I would love to meet (and even have lunch with)…."

"If I could have any superpower, it would be…."

"My perfect day would start with _____ and end with _____."

"If I could live anywhere in the world, it would be…."

"The most embarrassing thing that happened to me at work was…."

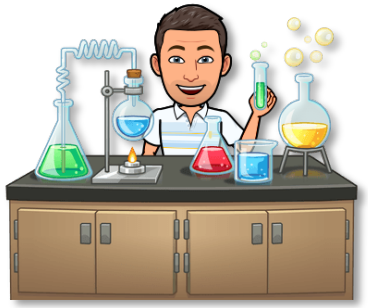
"I know it's crazy, but I love to eat…"

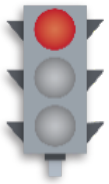
"If I won the lottery, the first thing I'd buy would be…."

"Instead of the job I have, I've always dreamed of being. . ."

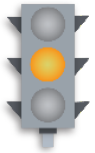
"Not many people know this about me, but when I was younger, I…"

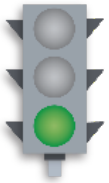
CLEAN TESTING





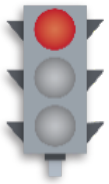
Tests are mostly done manually; the team struggles to tests some part of the code due to how the code is designed.



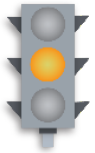


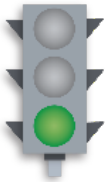
The team writes good unit tests and uses practices like TDD when appropriate.
Most of the code base is tested as soon as it's pushed to the centralized repository.

CLEAN CODE



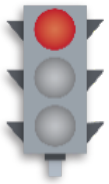
There is no alignment on what is a good piece of code in the team.
The code is difficult to maintain.



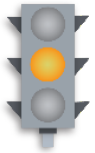


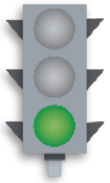
The code is simple and elegant.
Dev practices are shared within the team and in accordance with the standards of the organization.

SOLUTION DESIGN



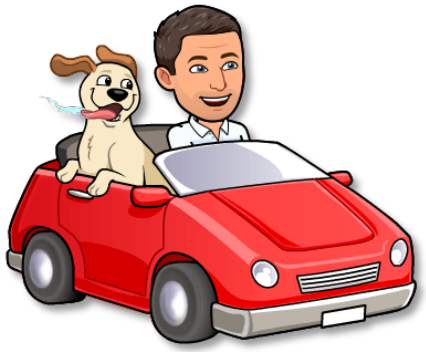
We work alone one our team topics and define our solutions by our own.

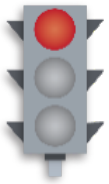




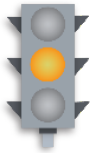
We collaborate with other people from the organization to design our solutions.
Our solutions are aligned with architecture guidelines.

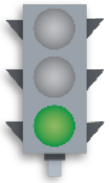
CONTINUOUS INTEGRATION





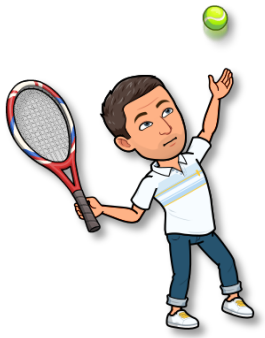
None or only part of our source code is built through a CI platform.

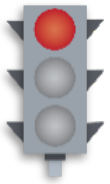




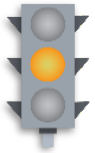
We build everything with our CI platform. In our process we use Code Analysis tools to get feedback on our code and improve its quality.

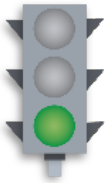
DEPLOYMENT





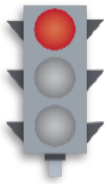
Integration problems are frequent, and deployments are done mostly manually.



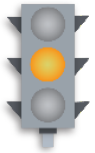


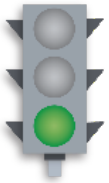
Everything is automated except the validations. The deployment pipeline increases the visibility, and the team does not fear to deploy in production as the issues are catch early.

APPLICATION MONITORING



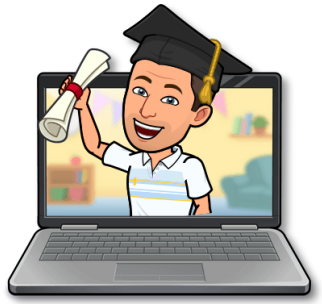
There is no log. When a problem occurred in production, we have no clue about what happened.

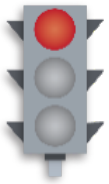




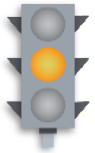
We understand logs without any pain. We can understand what happens on our system in only a few seconds.

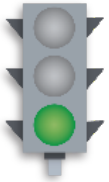
SKILLS & TRAINING





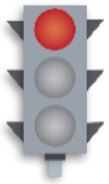
The team does not have all the necessaries skills to carry out the topics or does not fully understand the tools/frameworks and practices they need to work with.



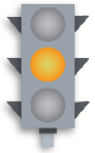


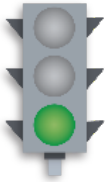
Team members have the skills to do their Job with confidence.
Tech watch and continuous learning are part of the team missions.

APPSEC



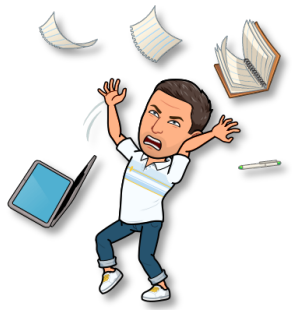
Application security is not included in the development cycle.
The team is not trained.

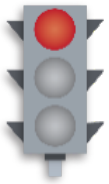




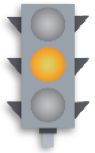
Security is part of the application's lifecycle : business abuse cases, static code analysis and pentests.
The team is trained and understand the importance of secure coding.

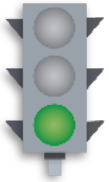
TECHNICAL DEBT MANAGEMENT





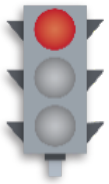
The technical debt increases uncontrollably either because of a lack of competence or because of the pressure placed on the team.



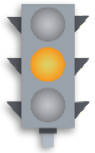


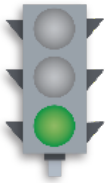
The team is concerned about quality and works daily to fight against technical debt. The PO is aware of the effort required.

COLLECTIVE OWNERSHIP



People are working in silos within the team. Some team members always work on the same type of tasks.





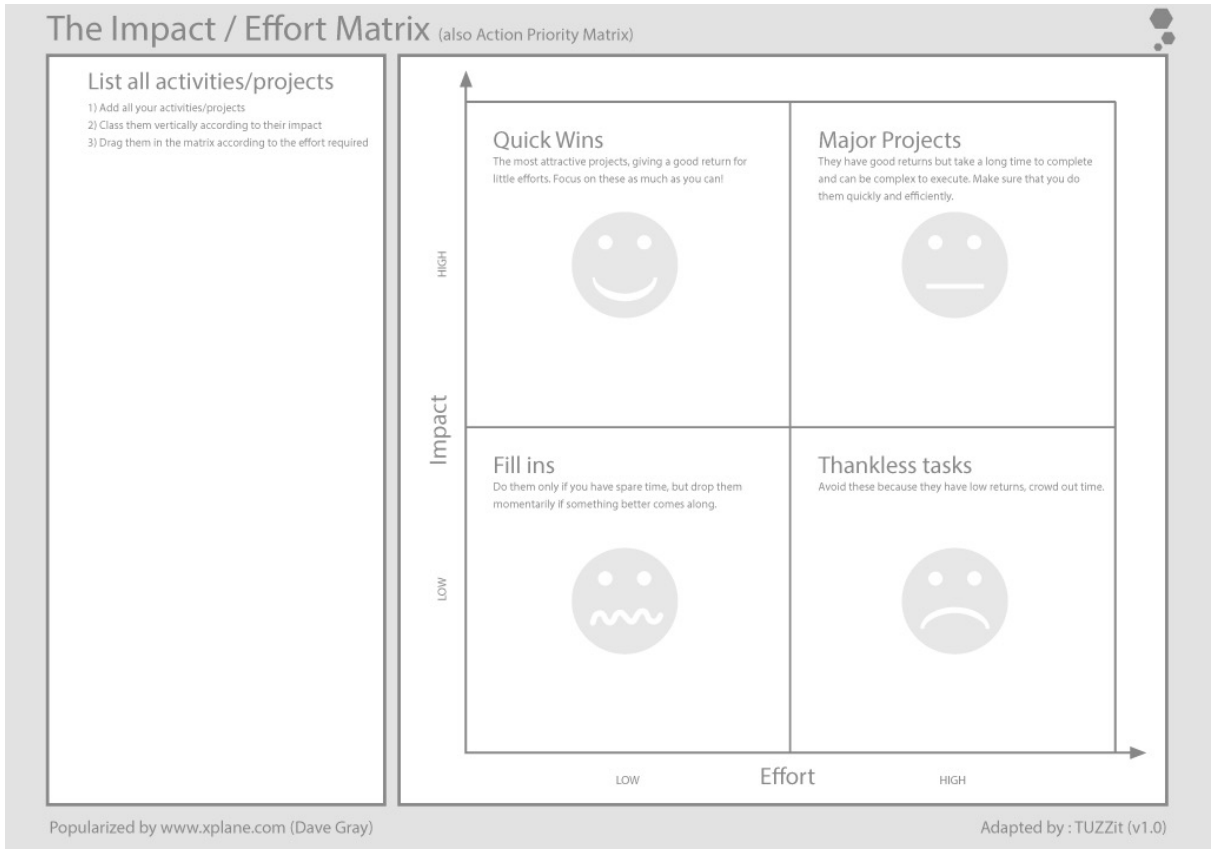
There is a culture of sharing in the team, Pair Programming and / or code review is a common practice.
When a build is broken there is no finger pointing and there is a shared responsibility to fix it.

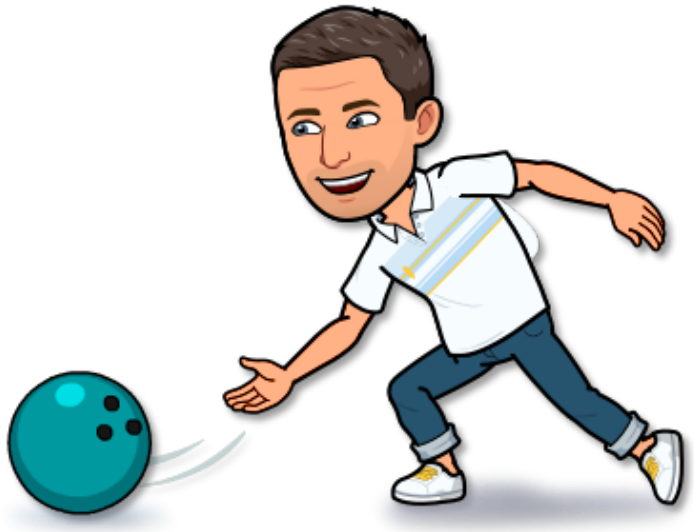
Tour dans le code

- Typical unit / integration test (if there are any)
- Any class or function that you feel is well designed and would like to see more like that
- A class or function that doesn't have any unit tests, which you would like to have tests for
- Any class or function that is notorious for having bugs in
- Any code that you've been working with recently that represents your current design thinking.

PRIORISATION AXES / ACTIONS

- MAPPER LES ITEMS SUR LA MATRICE IMPACT / EFFORT
- TIMELINE SOUHAITÉE





R.O.T.I

1

A WASTE OF MY TIME

2

MY TIME INVESTED EXCEEDED MY RETURN

3

I BROKE EVEN

4

I GAINED MORE THAN THE TIME I SPENT

5

REALLY USEFUL EVENT THAT WORTH MORE THAN THE TIME I SPENT ON IT

@YOT88