JOCHEN CHRIST

Teamwork im Home-Office

Remote Mob Programming

INNOQ

Simon Harrer

Senior Consultant
INNOQ

Jochen Christ

Senior Consultant
INNOQ

Martin Huber

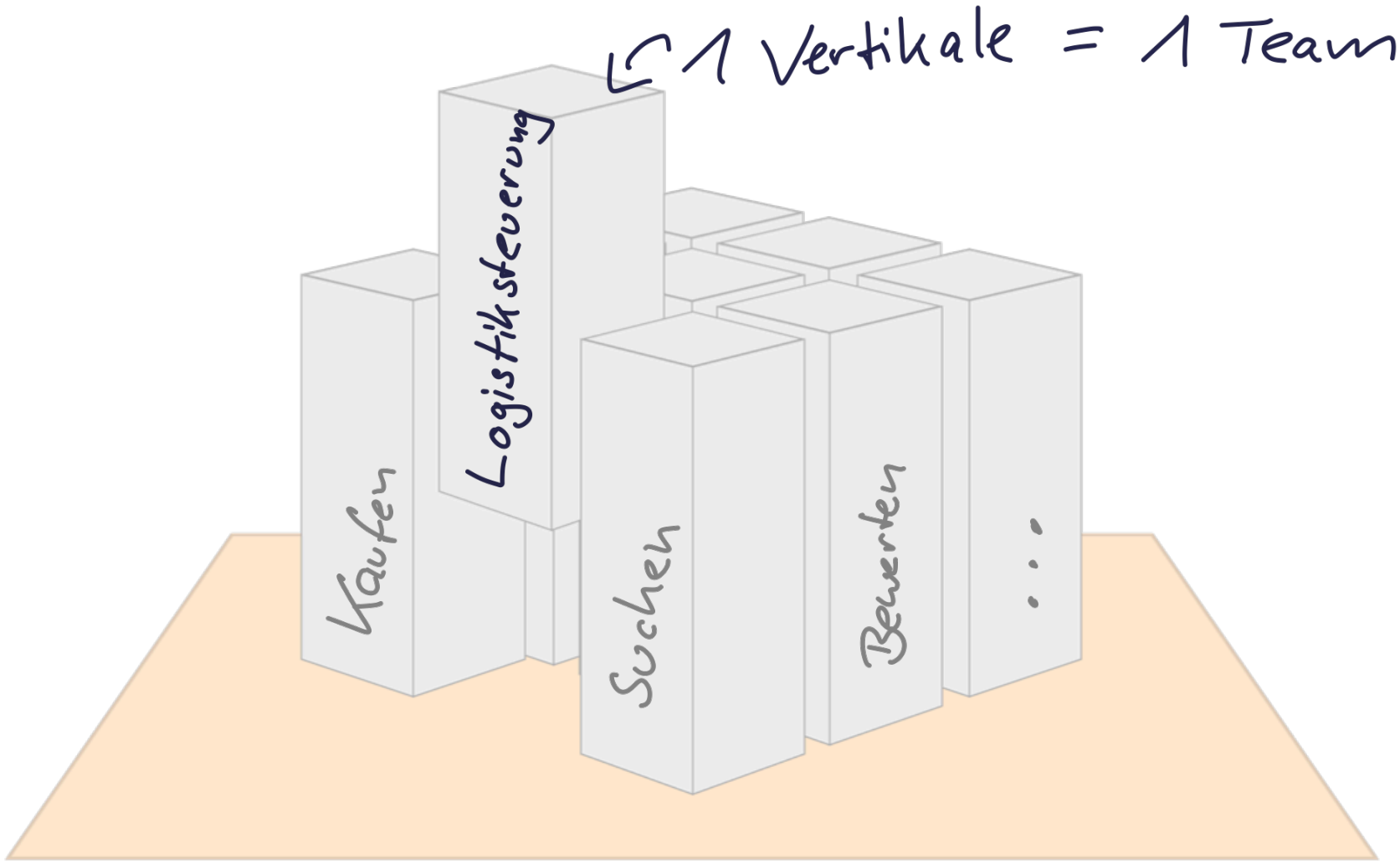
Senior Consultant
INNOQ

Simon Harrer
@simonharrer

I have been doing remote #mobprogramming full-time for a year now. One year without daily stand-ups. One year without waiting for reviews. One year without handovers before holidays. One year at home without feeling alone. I don't want to work differently anymore.

♡ 543 8:31 PM - Aug 12, 2019

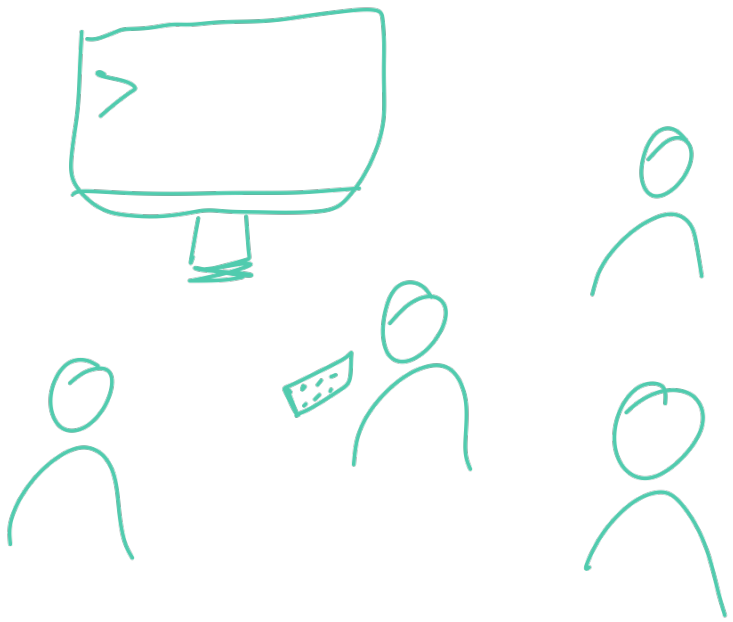
206 people are talking about this >



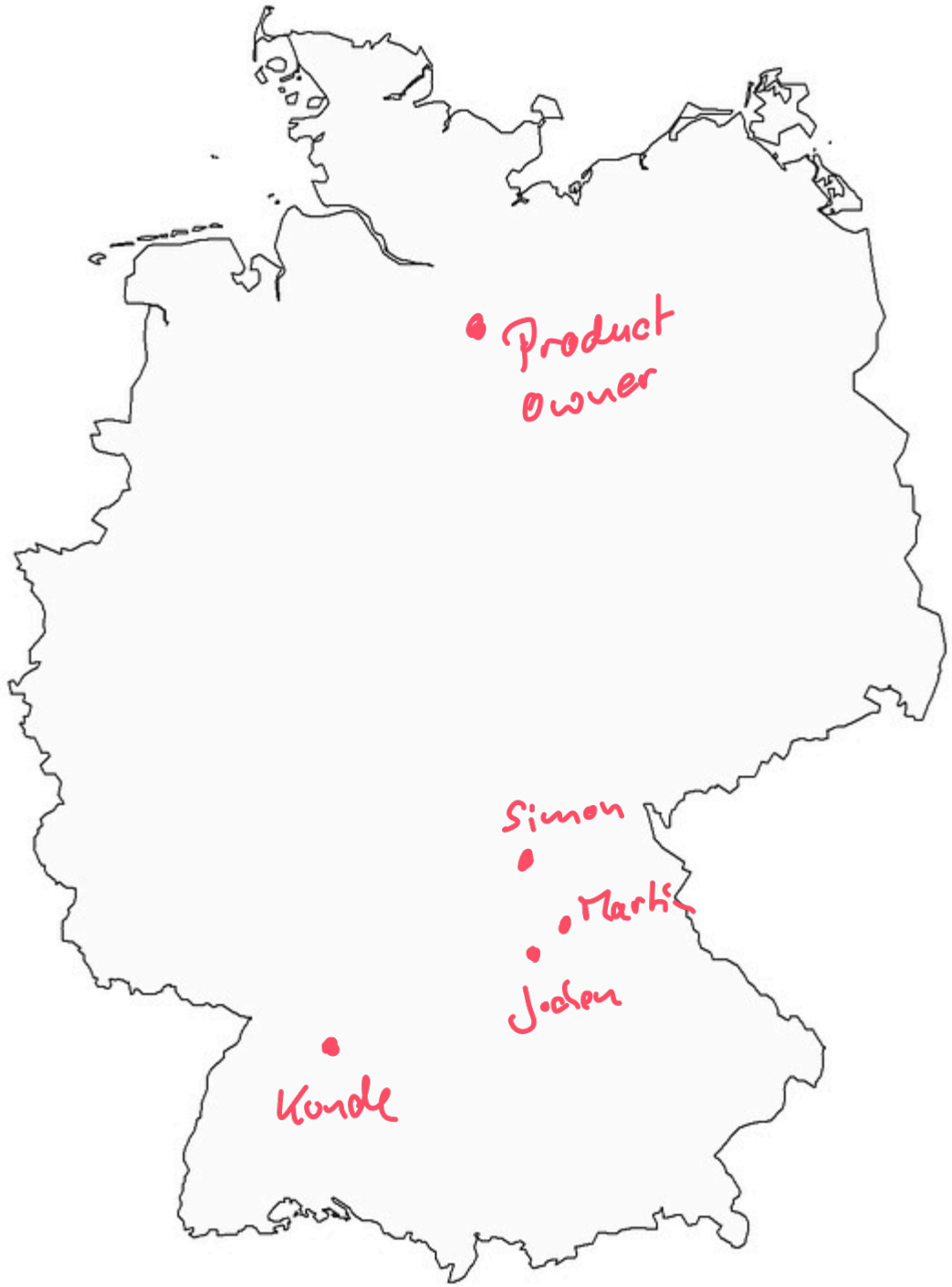
Das Gesamtsystem besteht aus eigenständigen fachlichen Vertikalen

Selbstorganisierende Teams

Kick-Off gemeinsam vor Ort



- Gemeinsam 2 Wochen
  - Fachlichkeit verstehen
  - Definition des Bounded-Context
  - Architektur und Technologie-Auswahl
- Unterschiedlicher Background und Know-How
  - => Erster Service gemeinsam begonnen

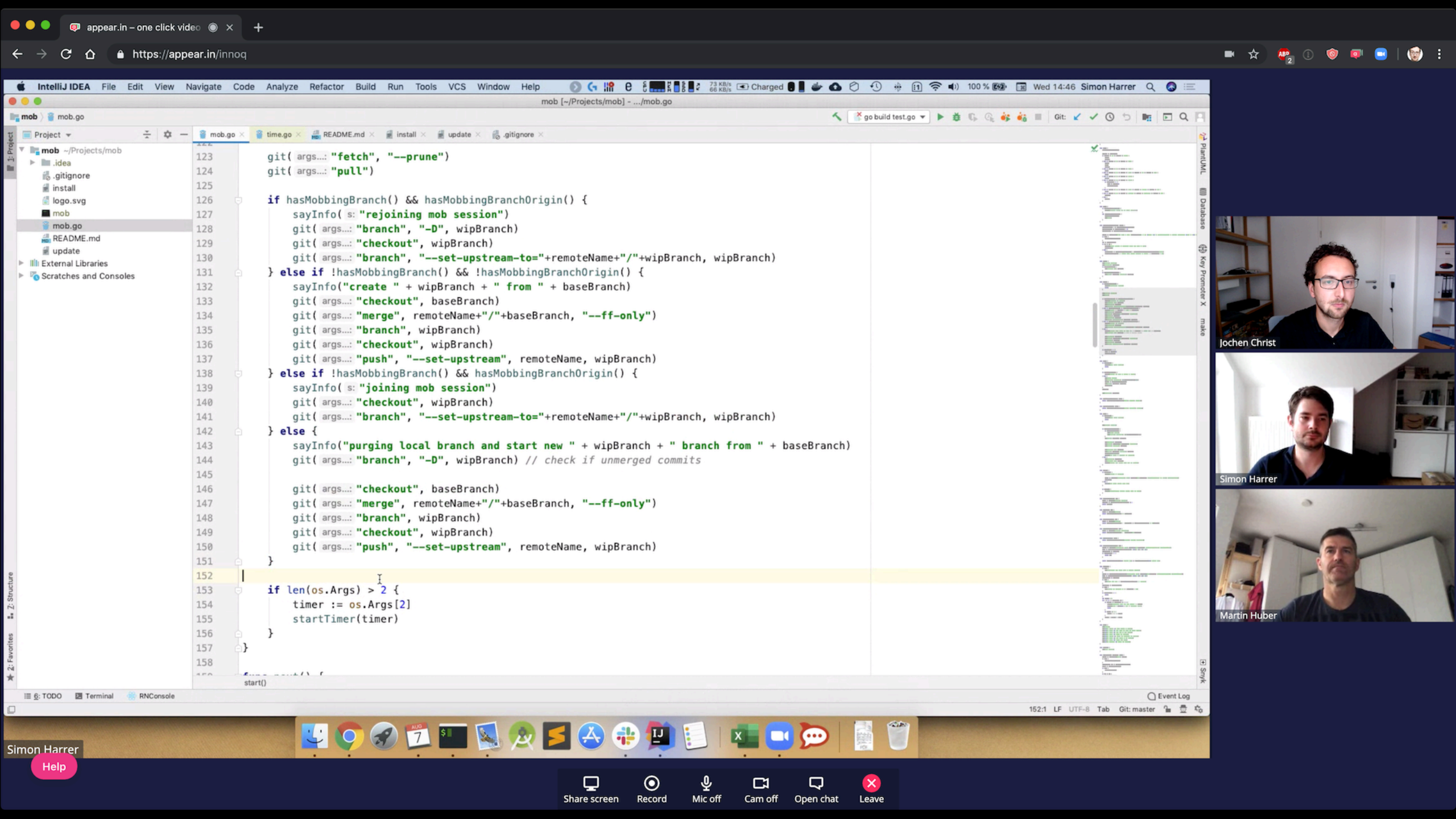

Product owner
Simon
Martin
Jochen
Kunde

Und dann?

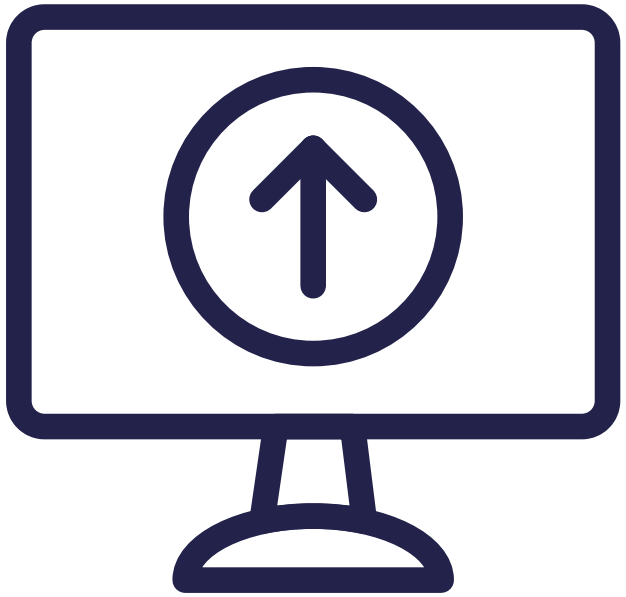
Wir wollten den begonnen Service gemeinsam fertig machen

... und sind bis heute in diesem Arbeitsmodus geblieben

Welcome to
Remote Mob Programming


appear.in – one click video
https://appear.in/innoq
IntelliJ IDEA File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
mob [~/Projects/mob] - .../mob.go
mob.go time.go README.md install update .gitignore
Project
mob ~/Projects/mob
.idea
.gitignore
install
logo.svg
mob
mob.go
README.md
update
External Libraries
Scratches and Consoles
git(args...: "fetch", "--prune")
git(args...: "pull")
if hasMobbingBranch() && hasMobbingBranchOrigin() {
sayInfo(s: "rejoining mob session")
git(args...: "branch", "-D", wipBranch)
git(args...: "checkout", wipBranch)
git(args...: "branch", "--set-upstream-to="+remoteName+"/"+wipBranch, wipBranch)
} else if !hasMobbingBranch() && !hasMobbingBranchOrigin() {
sayInfo("create " + wipBranch + " from " + baseBranch)
git(args...: "checkout", baseBranch)
git(args...: "merge", remoteName+"/"+baseBranch, "--ff-only")
git(args...: "branch", wipBranch)
git(args...: "checkout", wipBranch)
git(args...: "push", "--set-upstream", remoteName, wipBranch)
} else if !hasMobbingBranch() && hasMobbingBranchOrigin() {
sayInfo(s: "joining mob session")
git(args...: "checkout", wipBranch)
git(args...: "branch", "--set-upstream-to="+remoteName+"/"+wipBranch, wipBranch)
} else {
sayInfo("purging local branch and start new " + wipBranch + " branch from " + baseBranch)
git(args...: "branch", "-D", wipBranch) // check if unmerged commits
git(args...: "checkout", baseBranch)
git(args...: "merge", remoteName+"/"+baseBranch, "--ff-only")
git(args...: "branch", wipBranch)
git(args...: "checkout", wipBranch)
git(args...: "push", "--set-upstream", remoteName, wipBranch)
}
if len(os.Args) > 2 {
timer := os.Args[2]
startTimer(timer)
}
}
start()
6: TODO Terminal RNConsole
Event Log
152:1 LF UTF-8 Tab Git: master
Jochen Christ
Simon Harrer
Martin Huber
Simon Harrer
Help
Share screen Record Mic off Cam off Open chat Leave

Screen Sharing



We feel most comfortable working in our own individual environment. It is where we are most productive.

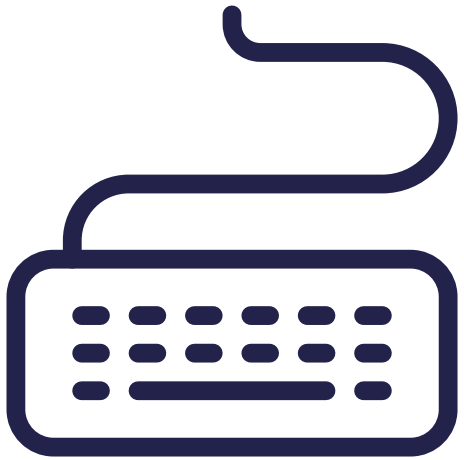
The typist shares their primary screen, showing the IDE.

We all look at the same shared screen.

It is highly efficient to work with actual code in contrast to having abstract meta discussions.

We share while working on Issues or Requirements.

Typist and the Rest of the Mob



Typist: One person controls the keyboard,
The rest of the mob: Discusses the problem, agrees on the solution, and instructs the typist.

We value the typist as they allow the rest of the mob to focus on solving the problem.

The typist must not code on their own.

Terminology is adopted from Code with the Wisdom of the Crowd by Mark Pearl.

10 Minutes Intervals

In a mob session, the typist role rotates periodically. Short rotation periods keep everyone concentrated and every opinion in the mix.

We rotate every ten minutes.

Surprisingly, taking your turn as a typist allows you a mental relaxation. You just wait for instructions.

Git Handover

With on-site Mob Programming, you just pass on the keyboard to hand over to the next person. This is a challenge for a distributed team.

We hand over with WIP commits on a temporary git branch.

To have a clean master branch, we work on a temporary mob-session branch.


```
➜  fizzbuzz git:(master) mob start 10
 ✓ [git fetch --prune]
 ✓ [git pull]
 > create mob-session from master
 ✓ [git checkout master]
 ✓ [git merge origin/master --ff-only]
 ✓ [git branch mob-session]
 ✓ [git checkout mob-session]
 ✓ [git push --set-upstream origin mob-session]
 ✓ 10 minutes timer started (finishes at approx. 18:41)
 > mobbing in progress

➜  fizzbuzz git:(mob-session) mob next
 ✓ [git add --all]
 ✓ [git commit --message "Mob Session DONE [ci-skip]"]
 ✓ [git push origin mob-session]
README.md | 2 ++
 1 file changed, 2 insertions(+)
 ✓ [git checkout master]
```

https://github.com/remotemobprogramming/mob

Remote Everybody

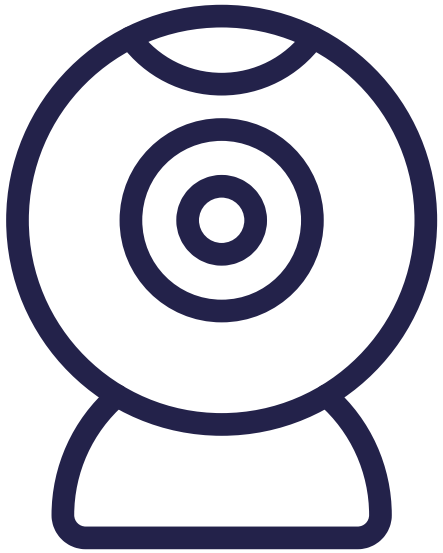
If one is remote, everybody is remote.

It does not work if part of the team works on-site.

This would lead to information asymmetry.

We all work from home, but don't feel alone.

In home office it is easier to communicate than in most open-plan offices.

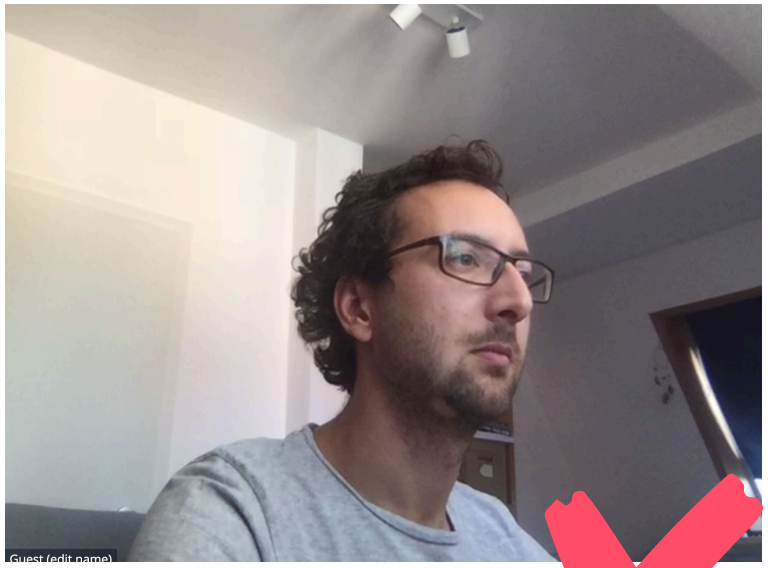


Camera Always On

Working face-to-face is powerful because we communicate with the whole body, not just our words.

We activate our cameras all the time.

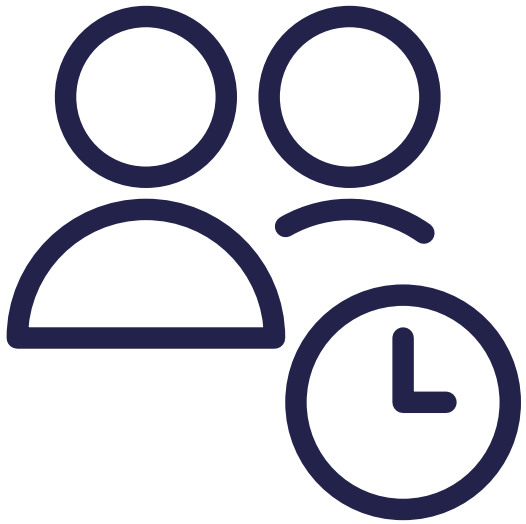
It gives a sense of presence in the team, almost like working in the same room together.


Guest (edit name)


Guest (edit name)


Guest (edit name)

Same Time



One of the prerequisites of Mob Programming is working at the same time.

We mob at least six hours a day.

We align our core working hours.
We also agree on the same lunch hour.

Small Team

The whole team works and focuses on the same thing.

Only one person can talk at the same time.

In larger teams it is harder to stay focused.

We are four.

The minimum size to form a mob is 3.
In our experience, teams with 3-4 developers provide the best benefit-cost ratio.

Group Decisions

Group decisions are superior over individual decisions. In Remote Mob Programming, all decisions are group decisions.

We minimize technical debt.

When we are coding, we all agree on changes and code style. As a consequence, we don't need code reviews or pull requests.

We document decisions with extensive consequences using Architecture Decision Records.

Idempotenz

Created by Jochen Christ, last modified on 27.09.2018

Die Kommunikation mit Umsystemen und innerhalb unserer Vertikale findet asynchron über Nachrichten statt.

Bei der Abwägung At-Least-Once-Delivery vs. At-Most-Once-Delivery wird At-Least-Once-Delivery gewählt, da wir keine Bestellungen verlieren dürfen. Es kann aus technischen Gründen daher vorkommen, dass Nachrichten mehrfach ankommen. Es muss verhindert werden, dass eine Bestellung mehrfach ausgeliefert wird.

Entscheidung

Alle Systeme verhalten sich idempotent.

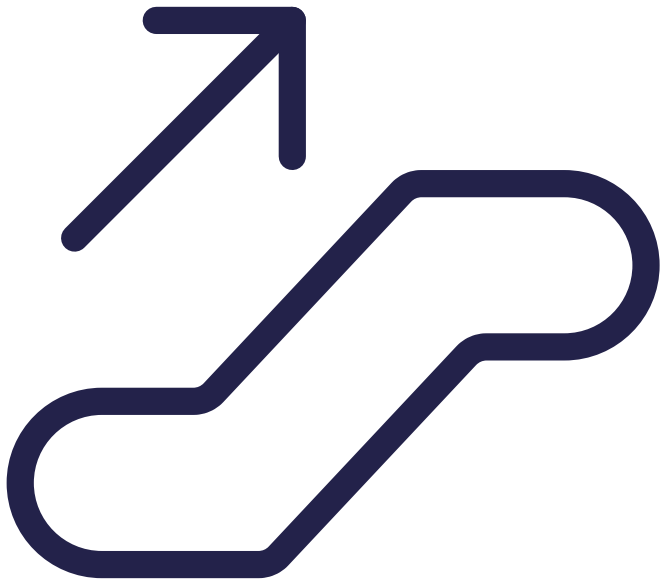
Status

Akzeptiert.

Konsequenzen

- Deduplication von Kafka-Nachrichten bzw. Erkennung von Duplikaten ist consumer-seitig notwendig.
- Producer können leichtgewichtiger implementiert werden.
- Alle Systeme müssen Tests auf idempotentes Verhalten beinhalten.

Constant Momentum



In a feature branch-based workflow, you are blocked waiting for the code review of your pull request.

While waiting, you start another feature and need to switch context.

As we aren't blocked by ourselves.

We have continuous and implicit code reviews
– no feature branches, no waiting, no context switches.

Learn from the Team

Sharing knowledge is at the heart of Mob Programming.

Everything the mob does is the result of in-depth discussions.
Nothing is done without agreeing on the why. That's where everybody learns.

We get better every day by learning from each other

With Mob Programming, on-boarding only takes weeks, not years.

Regular On-Site Meetings

The better everybody knows each other, the better everybody can collaborate remotely.
Getting to know each other works best on-site.

We meet on-site once a month.

On-site, we do: Long-term Plannings, Retrospectives, Whiteboard Sessions, Fun.

Trust

Our client does not see us working. So, management has a natural fear of losing control over the team.

Also, there is an inherent doubt of the team's productivity with everyone working one the same issues.

We build trust by actively communicating.

We write daily check-ins in our team's chat channel.

Favoriten

[redacted]

pruefen-best

pruefen-best-alerts-dev

pruefen-best-alerts-prod

pruefen-best-checkins

pruefen-best-intern

Unterhaltungen

Kanäle

[redacted]

breuninger

pruefen-best-checkins

Simon Harrer @exsharrer BEST PRFN 16:43

Heute eine tolle Diskussion mit Seb gehabt zum Thema "Reservierung - Eigener Bounded Context: ja/nein/vielleicht". Ergebnis hier https://confluence.br3uninger.de/display/PRUF/Reservierung+in+eigenem+Kontext nachlesbar. Es bleibt vorerst bei einem eigenen Context, und wir bewerten die Situation neu wenn wir evtl. Umlagerungen anstoßen müssen und dadurch eine Abhängigkeit auf den DOM Context bekommen.

Dabei haben wir beschlossen, bei der Reservierung den reservierung-feed über zwei JSON-Home Einträge erreichbar zu machen, einen für den reservierungsfortschritt und einen für die anforderung. Beide Einträge zeigen aber auf die gleiche Feed Instanz. Das ermöglicht uns später mal, diese Feeds aufzuteilen und ergibt ein klareres Bild nach außen für die anderen Vertikalen (CTRL interessiert sich nur für den Reservierungsfortschritt, SFS interessiert sich nur für erstellte oder stornierte Anforderungen).

In eigener Sache: bald erscheint ein kleines Büchlein zu Remote Mob Programming https://leanpub.com/remotemobprogramming

leanpub.com

Remote Mob Programming

Remote Mob Programming combines two ways of working: Mob Programming and working as a distributed team. J…

Martin Huber @exmhuber BEST PRFN Owner 17:42

heute

...prinzipiell nur noch zu zweit, haben Christoph und ich uns den Fullload des AX-Bestandes vorgenommen und die Verbindung zum FTP-Server getestet und nachgewiesen und überlegt ob und wie wir einen IntegrationTest machen könnten. Wir haben uns dann auf aufgrund erhöhter Komplexität dafür entschieden auf einen IntegrationTest zu verzichten und nur die tatsächliche (Fach-)Logik zu testen. Dazu müssen wir die aktuelle Implementierung zwar etwas refaktorieren, sollte aber weniger Aufwand sein, als den IntegrationTest zu bauen um nachzuweisen, dass die Frameworks die wir dafür verwenden das können was sie versprechen 😉

Morgen geht es dann mit diesem Thema und Teilaufgaben aus dem Reservierungskontext weiter.

Nachricht

B i S </> \[KaTeX\]

Save the Planet

Daily commuting causes traffic jams, crowded trains, and significant greenhouse gas emissions. Even worse, some consultants fly to their customers' offices.

We don't travel, so zero greenhouse gas emissions.

No travel means no travel costs for us and our customers.

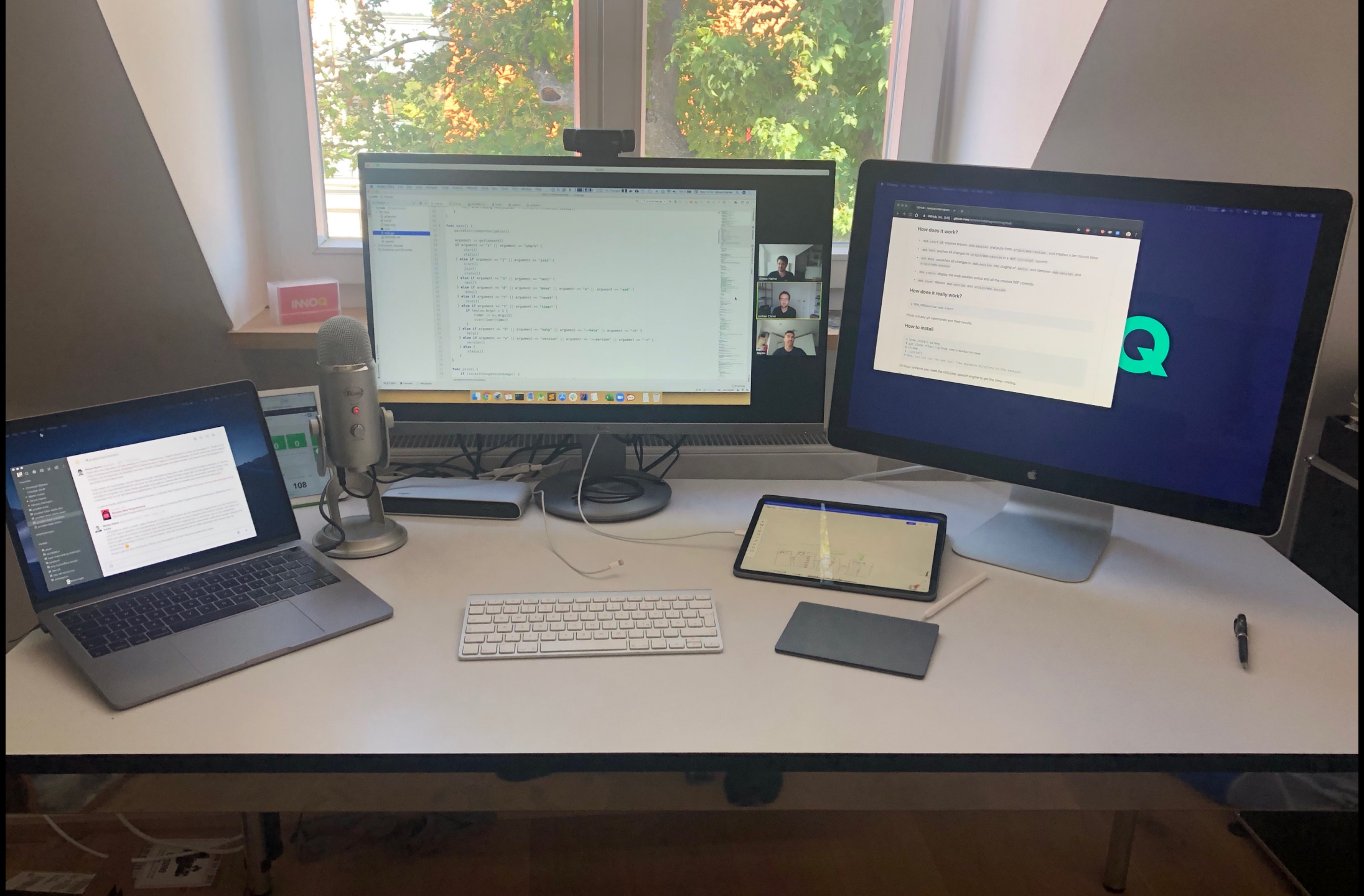
Dine with your Family

As software engineers, we often struggle to balance challenging and rewarding work with time for family and leisure.

We enjoy more quality time with our families.

With Remote Mob Programming we get both, rewarding work and quality time with our families and kids.

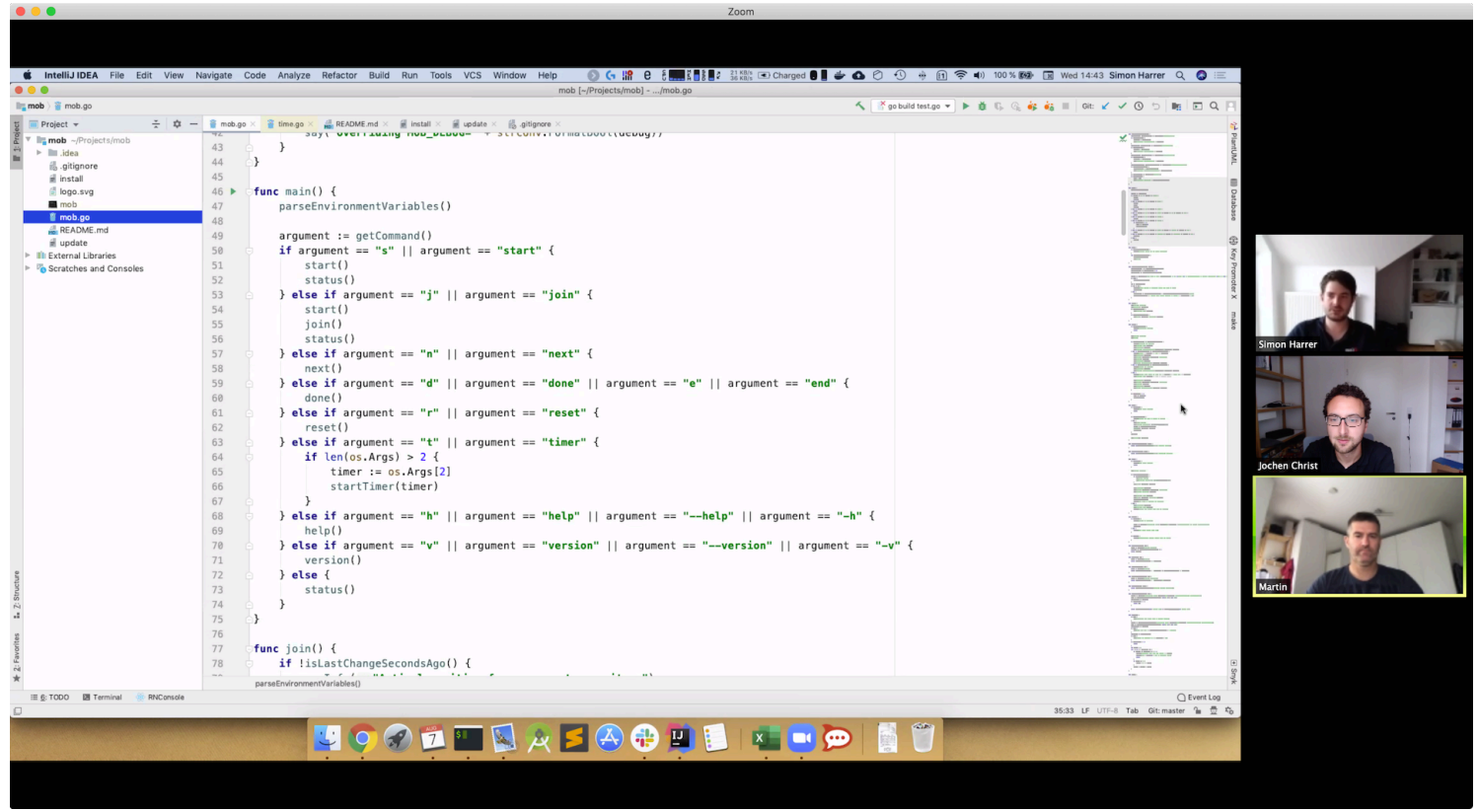
Our Setup


INNOQ
How does it work?
How does it really work?
How to install


sayInfo("create " + wipBranch + " from " + baseBranch)
git(args...: "checkout", baseBranch)
git(args...: "merge", remoteName+"/"+baseBranch, "--ff-only")
git(args...: "branch", wipBranch)
git(args...: "checkout", wipBranch)
git(args...: "push", "--set-upstream", remoteName, wipBranch)
} else if !hasMobbingBranch() && hasMobbingBranchOrigin() {
sayInfo(s: "joining mob session")
git(args...: "checkout", wipBranch)
git(args...: "branch", "--set-upstream-to="+remoteName+"/"+wipBranch
else {
sayInfo("purging local branch and start new " + wipBranch + " bra
git(args...: "branch", "-D", wipBranch) // check if unmerged commits
git(args...: "checkout", baseBranch)
git(args...: "merge", remoteName+"/"+baseBranch, "--ff-only")
git(args...: "branch", wipBranch)
git(args...: "checkout", wipBranch)
git(args...: "push", "--set-upstream", remoteName, wipBranch)
(os.Args) > 2 {
imer := os.Args[2]
tartTimer(timer)
Share screen
Record
Mic off
MUTE
VOLUME
108

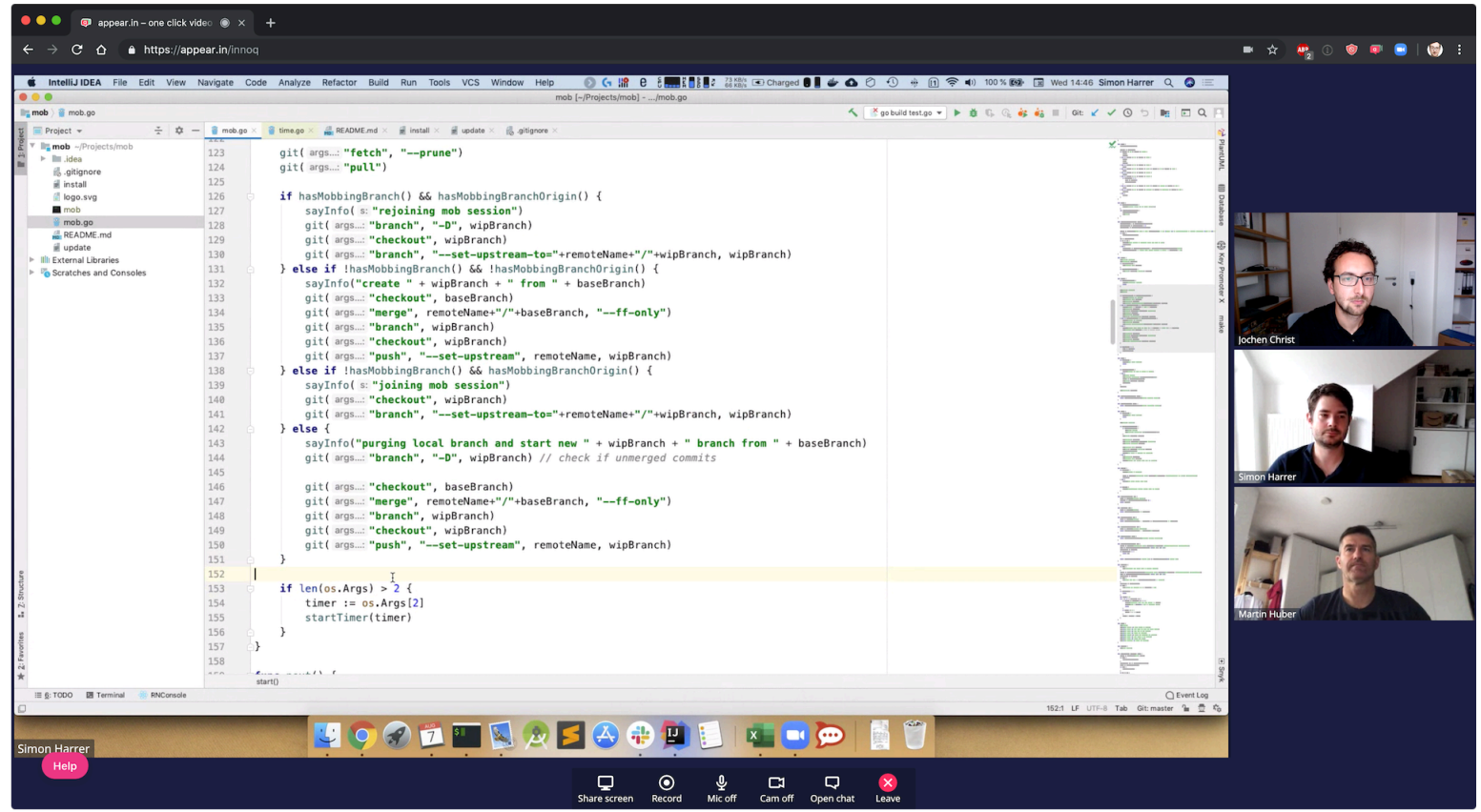

HD 1080p
logi

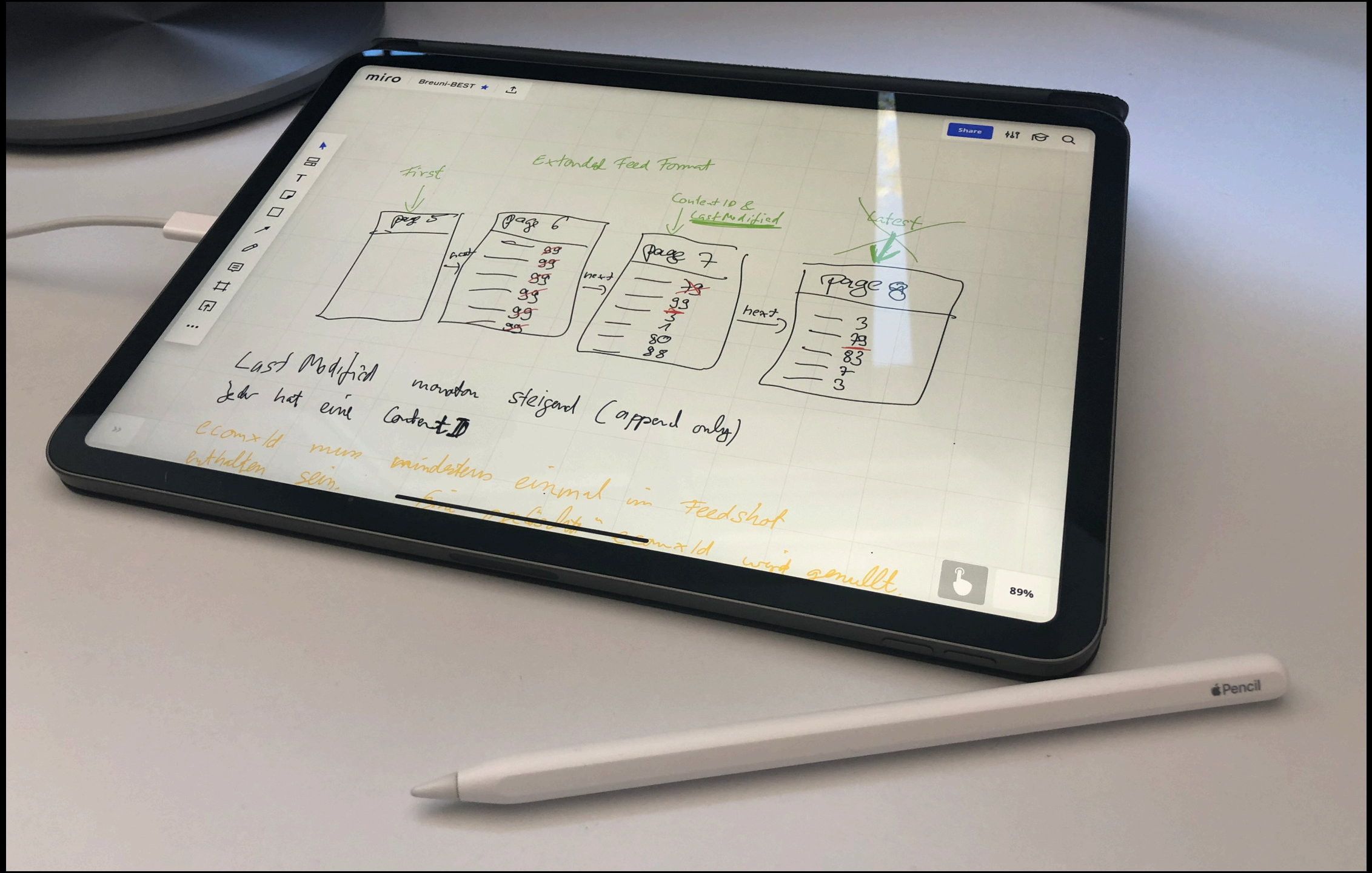

zoom


Zoom
IntelliJ IDEA File Edit View Navigate Code Analyze Refactor Build Run Tools VCS Window Help
mob [~/Projects/mob] - .../mob.go
mob.go
time.go
README.md
install
update
.gitignore
Project
mob ~/Projects/mob
.idea
.gitignore
install
logo.svg
mob
mob.go
README.md
update
External Libraries
Scratches and Consoles
func main() {
parseEnvironmentVariables()
argument := getCommand()
if argument == "s" || argument == "start" {
start()
status()
} else if argument == "j" || argument == "join" {
start()
join()
status()
} else if argument == "n" || argument == "next" {
next()
} else if argument == "d" || argument == "done" || argument == "e" || argument == "end" {
done()
} else if argument == "r" || argument == "reset" {
reset()
} else if argument == "t" || argument == "timer" {
if len(os.Args) > 2 {
timer := os.Args[2]
startTimer(timer)
}
} else if argument == "h" || argument == "help" || argument == "--help" || argument == "-h" {
help()
} else if argument == "v" || argument == "version" || argument == "--version" || argument == "-v" {
version()
} else {
status()
}
}
func join() {
if !isLastChangeSecondsAgo() {
parseEnvironmentVariables()
TODO Terminal RNConsole
Event Log
Simon Harrer
Jochen Christ
Martin

Whereby

(was: appear.in)




miro
Breuni-BEST
Share
Extended Feed Format
First
Page 5
Page 6
Page 7
Page 8
next
next
next
Content ID &
Last Modified
Latest
Last Modified
Content ID
steigend (append only)
89%
Pencil

Main Benefits

Time to market

Knowledge Sharing

Simon Harrer @simonharrer · Aug 12, 2019

I have been doing remote #mobprogramming full-time for a year now. One year without daily stand-ups. One year without waiting for reviews. One year without handovers before holidays. One year at home without feeling alone. I don't want to work differently anymore.

Claudio Zizza
@SenseException

This must be this "teamwork" everyone is talking about. It is even literally team work.

♡ 2 11:00 AM - Aug 13, 2019

See Claudio Zizza's other Tweets

www.remotemobprogramming.org

Free Booklet:
leanpub.com/remotemobprogramming

JOCHEN CHRIST

Remote Mob Programming

INNOQ