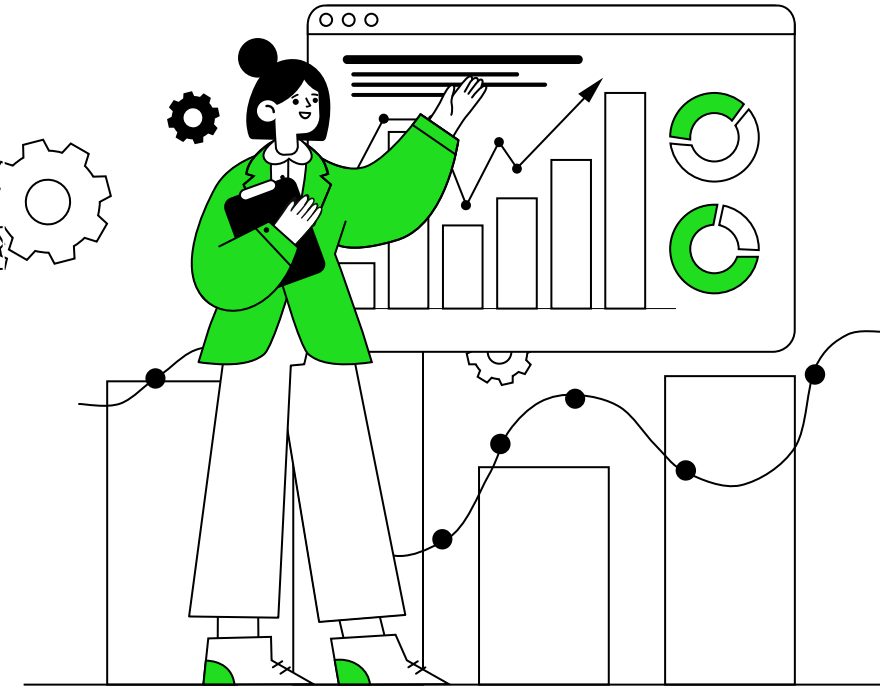
# HOW TO RELEASE A NEW KUBERNETES VERSION

Max Körbächer



**Max Körbächer**
Co-Founder & Manager Platform Engineering @ Liquid Reply

Part of the Kubernetes Release Team & Release Engineering since v1.17

mkoerbaecher

mkoerbi

# NEXT STOPS

01

## KUBERNETES

A project, a community, a whole universe

02

## FROM A COMMIT TO A RELEASE

Overview of the release process

03

## THE FIRST PR

System side tests

04

## FROM THE MERGE INTO MAIN

7060 tests later

05

## THE RELEASE

beta.1, RC.1, a new major version

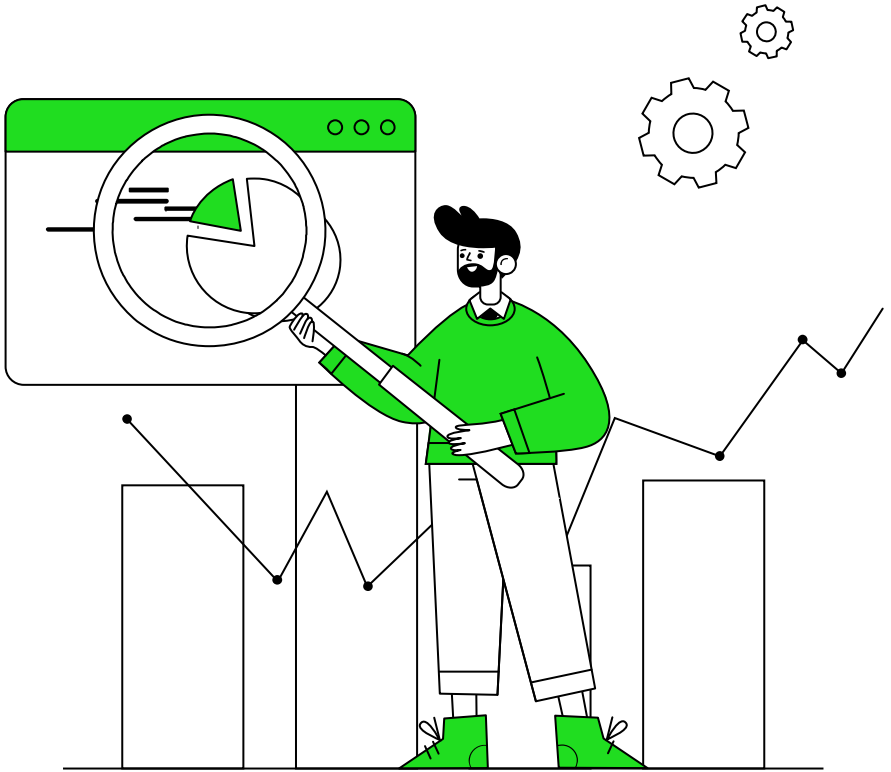
06

## LESSONS LEARNED

What you can take away from the K8s release!

01

# WHAT IS KUBERNETES



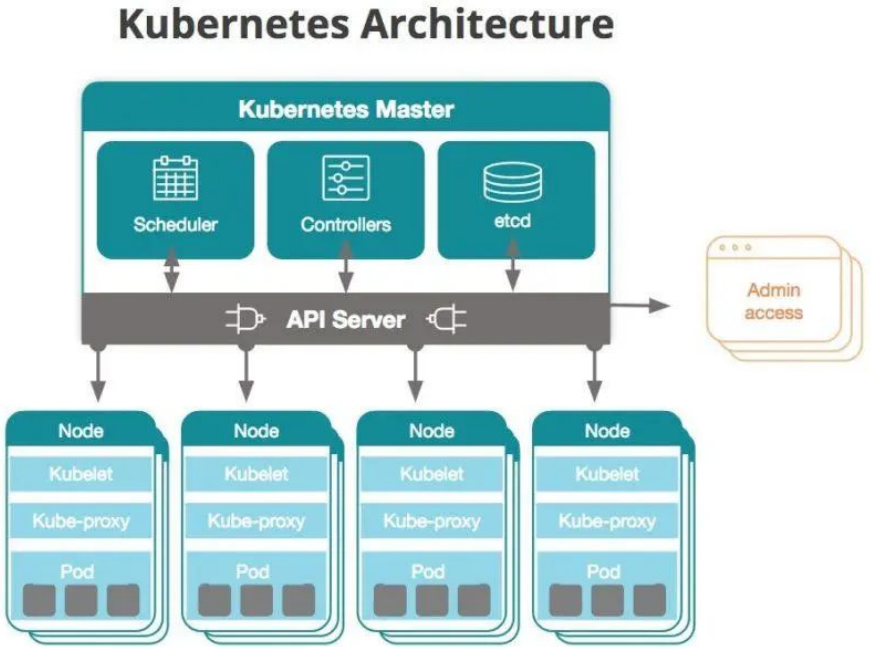
Solved all your problems. You're welcome.

# KUBERNETES

Kubernetes supports the deployment and operation of applications with a micro service architecture.

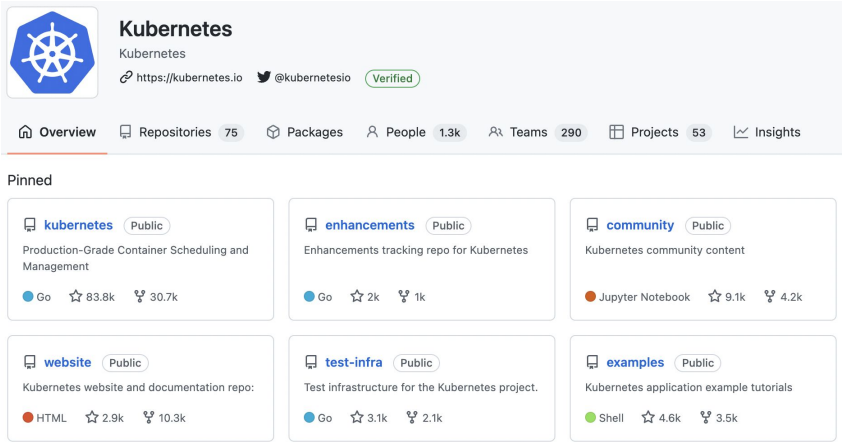
This is done through an abstraction layer across a group of nodes which handles containers.

Developers can deploy their containerized applications towards K8s while K8s takes care about basic operations and keep alive & scaling mechanisms.




Source: https://newrelic.com/blog/how-to-relic/what-is-kubernetes

# KUBERNETES PROJEKT





| Metric name | Metric value |
|---|---|
| Code committers | 1134 |
| Commenters | 4862 |
| Comments | 67890 |
| Commit commenters | 22 |
| Commits | 8981 |
| Contributions | 81826 |
| Contributors | 5659 |
| Events | 229185 |
| Forkers | 4448 |
| Issue commenters | 4755 |
| Issue creators | 2106 |
| Issues | 4304 |
| PR creators | 1540 |
| PR reviewers | 1003 |
| PRs | 8993 |

# 02

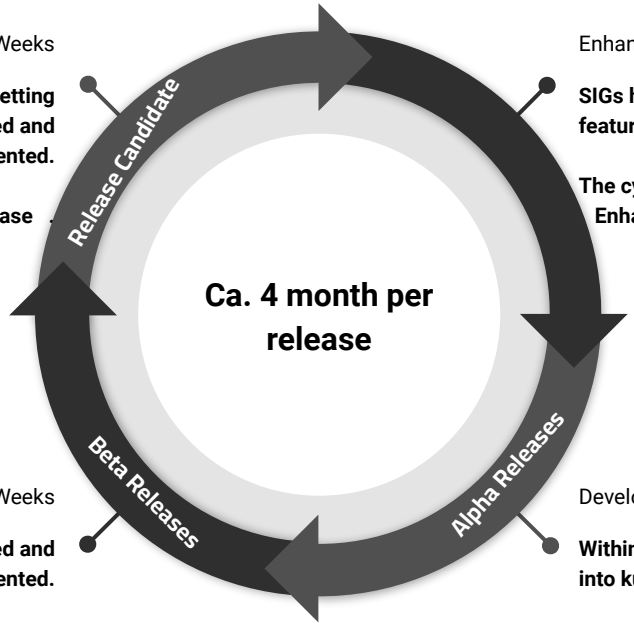
# FROM COMMIT
# TO THE RELEASE

# RELEASE CYCLE

Release Preparation & Testing - 2 Weeks

**Within the final cycle the new release getting prepared, a new release branch gets created and required jobs will be implemented.**

**This cycle ends with the major Release .**




Enhancement Cycle - 3 Weeks

**SIGs have to opt-in their to be released feature**

**The cycle ends with an Enhancement Freeze**

Beta Testing - 1 Weeks

**Final tests needs to be adjusted and implemented.**

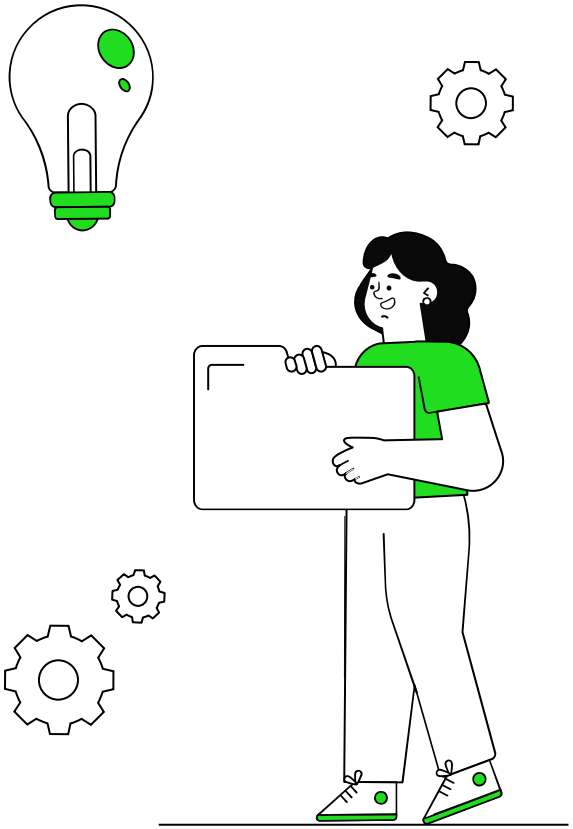
**The cycle ends with a Test Freeze**

Development Cycle - 10 Weeks

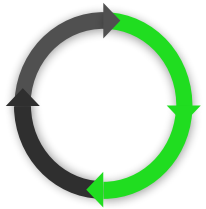
**Within 10 weeks the SIGs have to merge the KEPS into kubernetes/kubernetes**

**The cycle ends with an Code Freeze**

# 03 THE FIRST PULL REQUEST



# FROM DEVELOPMENT TO PR

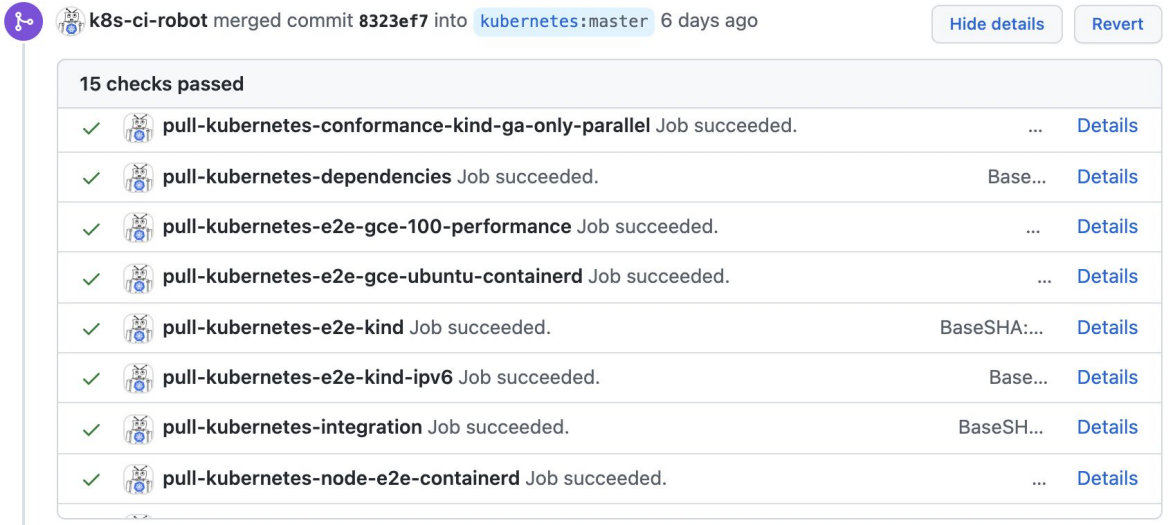


The development happens locally, all test can and must be executed once before committed.

As soon as the developer decides to move on with an PR, PR pre-submit tests and checks are automatically executed.

K8s test infrastructure:

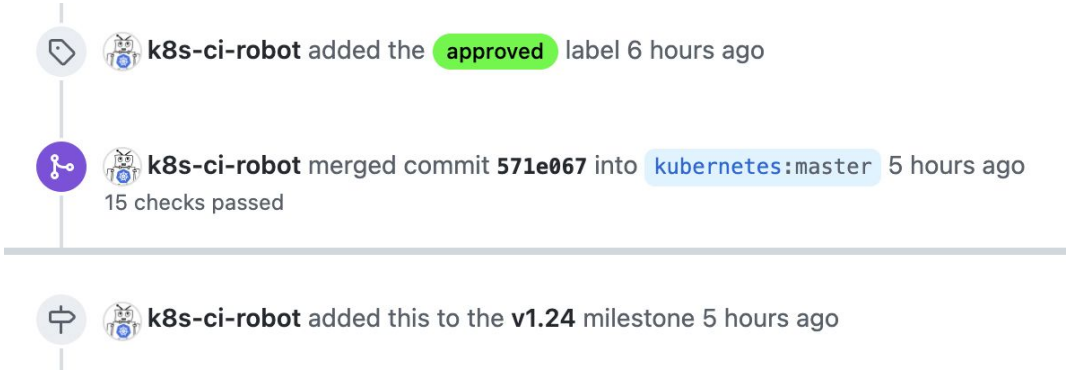
https://github.com/kubernetes/test-infra



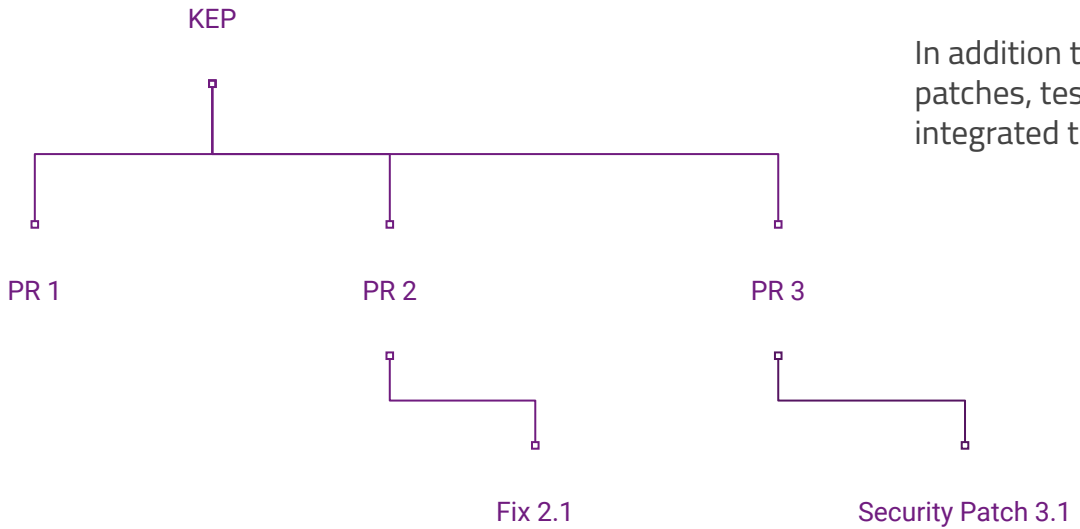
# TAKING OVER A COMMIT INTO THE PLANNED RELEASE

If a commit is merged into the main branch, it gets labeled with a milestone.

Milestones create a mapping between the opt-in KEP and the truly implemented feature.



# A KEP CAN BE ONE OR MANY PR




KEPs are not obviously visible. That's also caused by the fact that a KEP normally consist of multiple PRs. In an enterprise this maybe comes close to an Epic.

In addition to KEPs, we also handle fixes, patches, tests and cherry picks, which are integrated through an fast forward push.

# 04

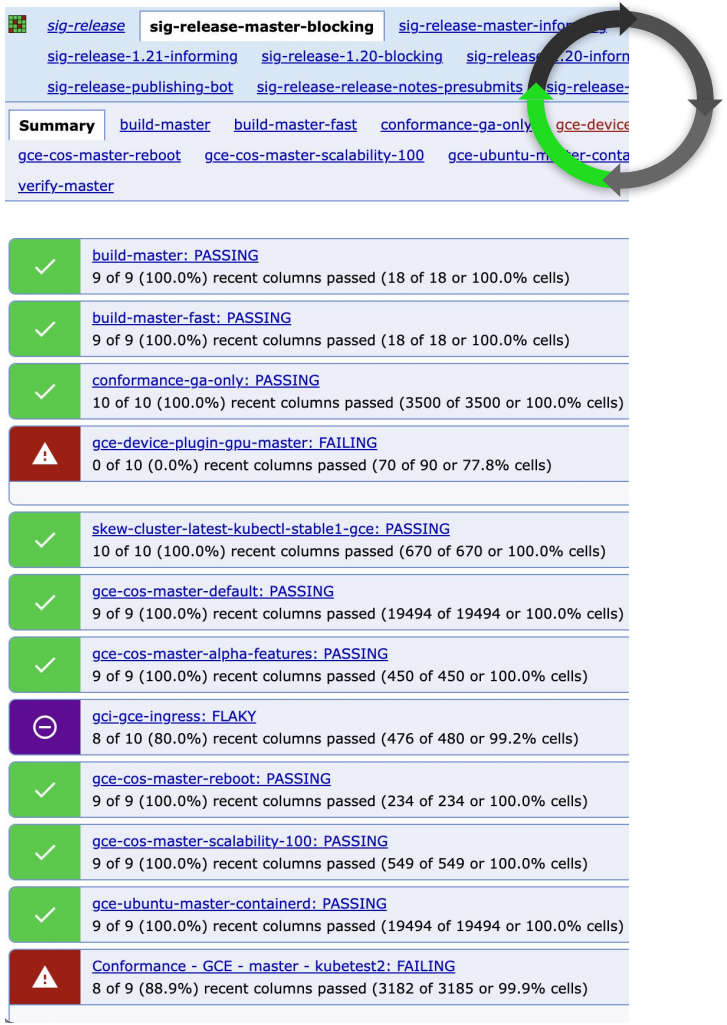
# AFTER THE MERGE

# CONTINUOUS TESTING

As soon as the PR is closed and the code is within the main branch - it will get tested within the test infrastructure.

Test are executed within different cycles, but at least once per day.

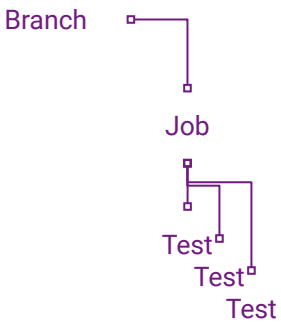
Tesgrid:
https://testgrid.k8s.io/sig-release-master-blocking



# TESTGRID


Tesgrid is a system that continuously scrapes the logs of Prow (CI/CD) and shows the results.

Tests are running specifically for the various 3 major versions, the master branch and are differentiated between informing and blocking.



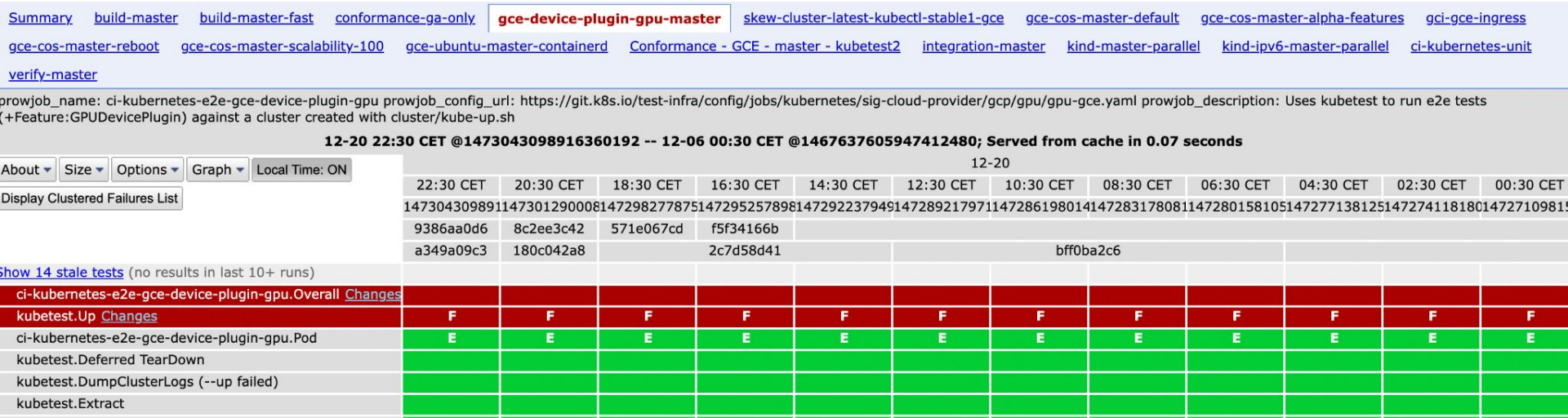

Each branch has therefore two tabs, within each tab are multiple jobs, clustering sometimes hundreds of tests.

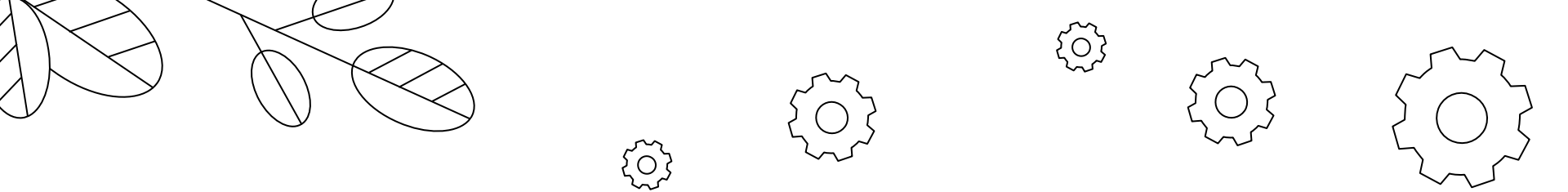
7044 tests at the moment!

# TESTGRID

Testgrid is a pool of information:

- Which test was executed when?
- Was the test successful or did it fail? Is it a flaky test?
- After which changes did the failure happen?
- Build number, K8s code commit hash, infrastructure hash
- Each box links to the prow & spyglass dashboard for exactly this test





# A short tour through Testgrid

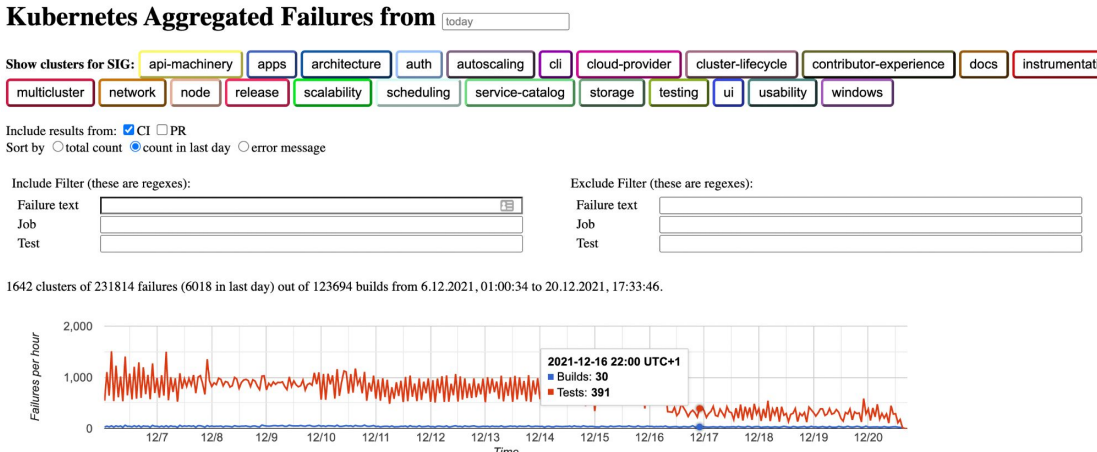
# TESTS ARE MULTI-DIMENSIONAL

The test is executed in different jobs. Therefore, a test can be seen multi-dimensional. That's because a job clusters a functionality, while for a functionality multiple components are required.

Testgrid is job oriented, to see the test perspective we have Triage tool in which you can research for the different test.

Example:
Scaling Test - executed at the Replica & load test, as well as for different rollout strategies.

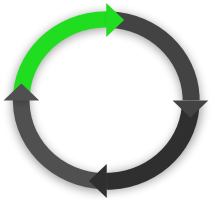
Here you can play around:
https://storage.googleapis.com/k8s-triage/index.html



05

# THE RELEASE

# PREPARATION AND EXECUTION

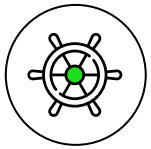


## BRANCHING

Split off of the release branche & e2e test of the new release version

## DOCUMENTATION

Publishing release notes, K8s documentation and communication are prepared



## RELEASE CUT

A Release Manager runs the krel tooling to create a release

# A OWN SET OF TOOLS WAS DEVELOPED FOR THE RELEASE CUT

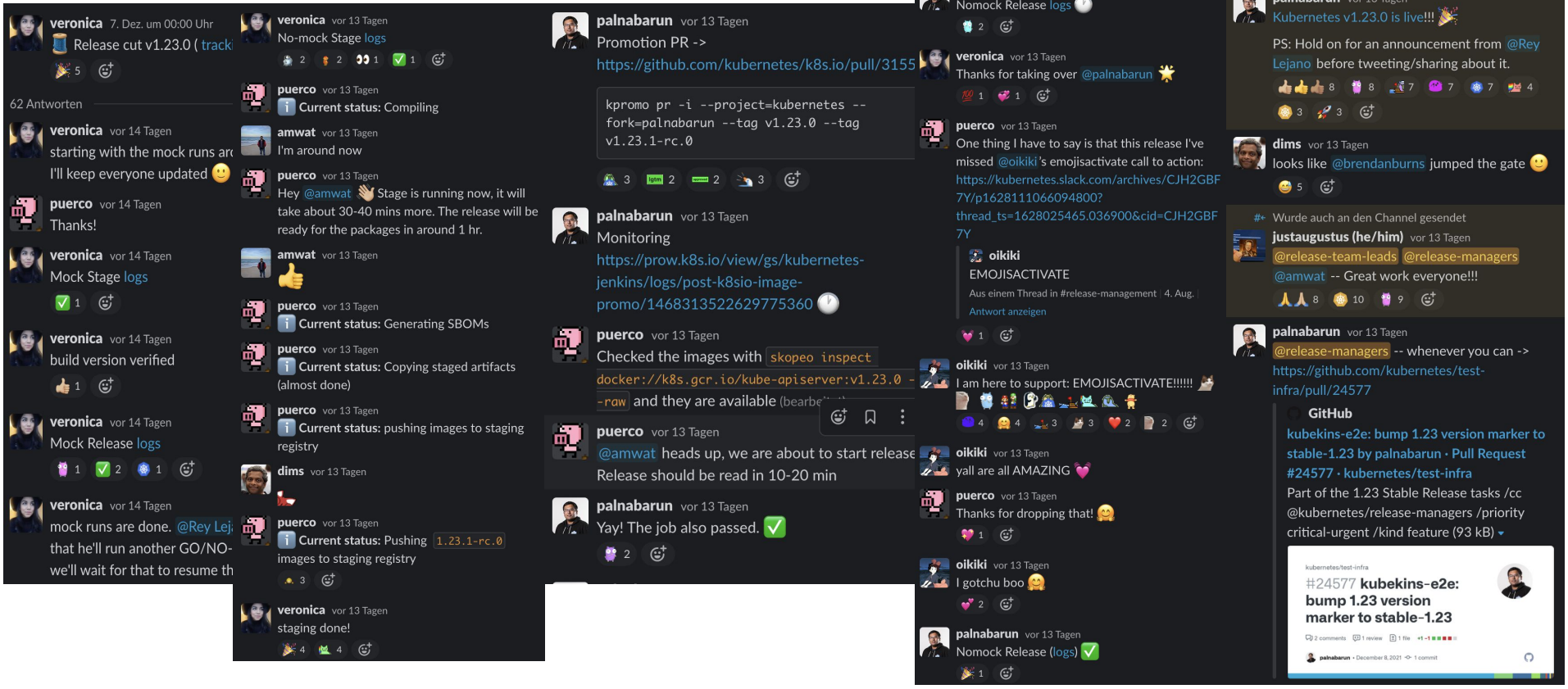
- Release Management
  - `krel`
  - `schedule-builder`
- Artifact Management
  - `kpromo`
  - `kubepkg`
  - `cip-mm`
  - `gh2gcs`
- End User
  - `bom`
  - `release-notes`
  - `gcbuilder`
  - `publish-release`
- Legacy
  - `push-build.sh`
- Contributing

**K**ubernetes **rel**ease Toolbox - KREL

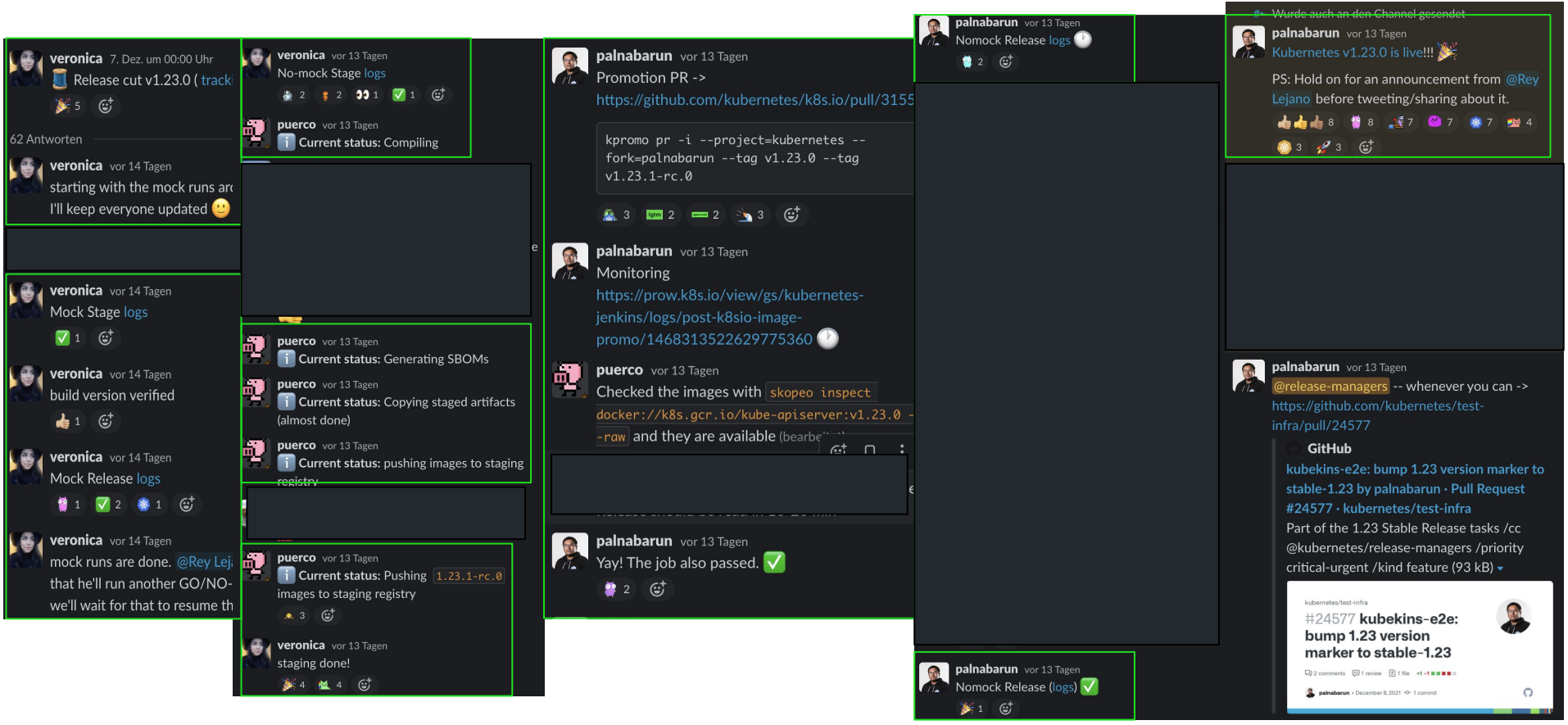
Krel is the central and critical component for the release cuts. The release cut is the step where within kubernetes/kubernetes main Branch a new release is created and packaged.

| Subcommand | Description |
|---|---|
| announce | Build and announce Kubernetes releases |
| changelog | Automate the lifecycle of CHANGELOG-x.y.{md,html} files in a k/k repository |
| ci-build | Build Kubernetes in CI and push release artifacts to Google Cloud Storage (GCS) |
| cve | Add and edit CVE information |
| ff | Fast forward a Kubernetes release branch |
| history | Run history to build a list of commands that ran when cutting a specific Kubernetes release |
| push | Push Kubernetes release artifacts to Google Cloud Storage (GCS) |
| release | Release a staged Kubernetes version |
| release-notes | The subcommand of choice for the Release Notes subteam of SIG Release |
| stage | Stage a new Kubernetes version |
| testgridshot | Take a screenshot of the testgrid dashboards |

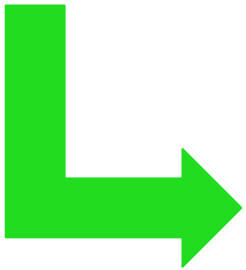
# THE RELEASE PROCESS USER PERSPECTIVE

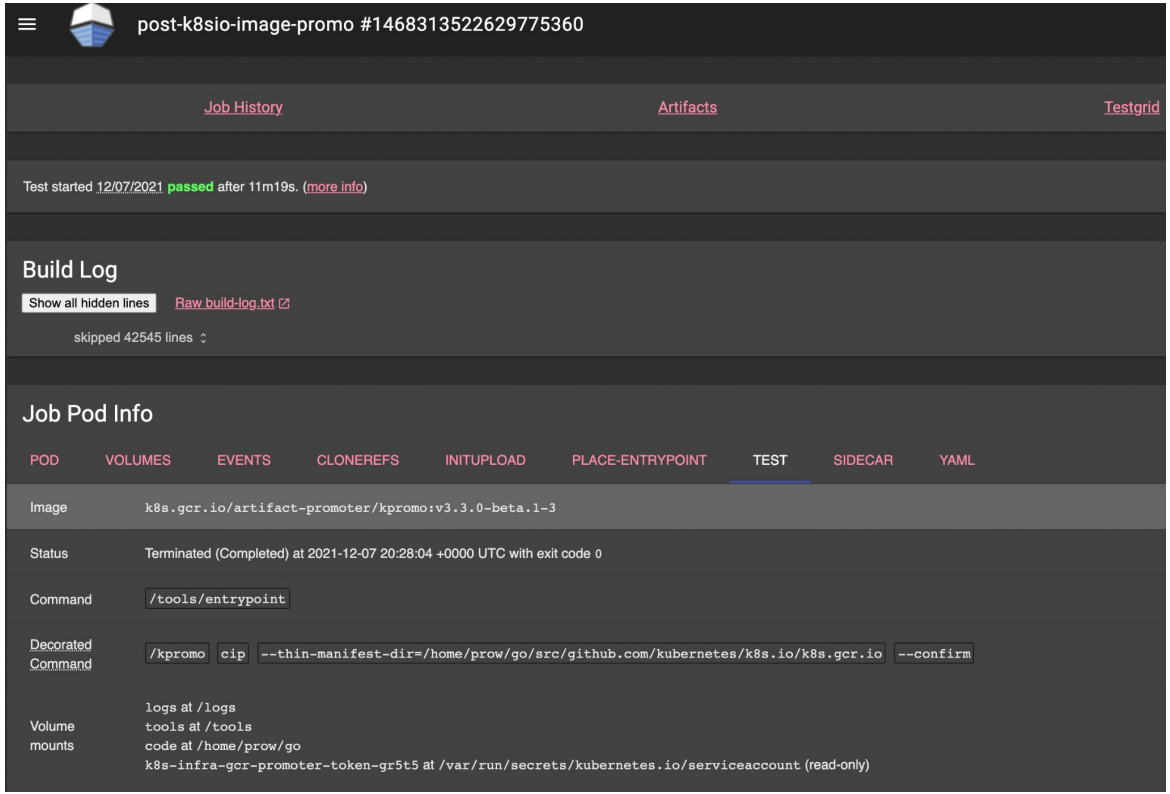


# ADDING SOME NOISE CANCELING

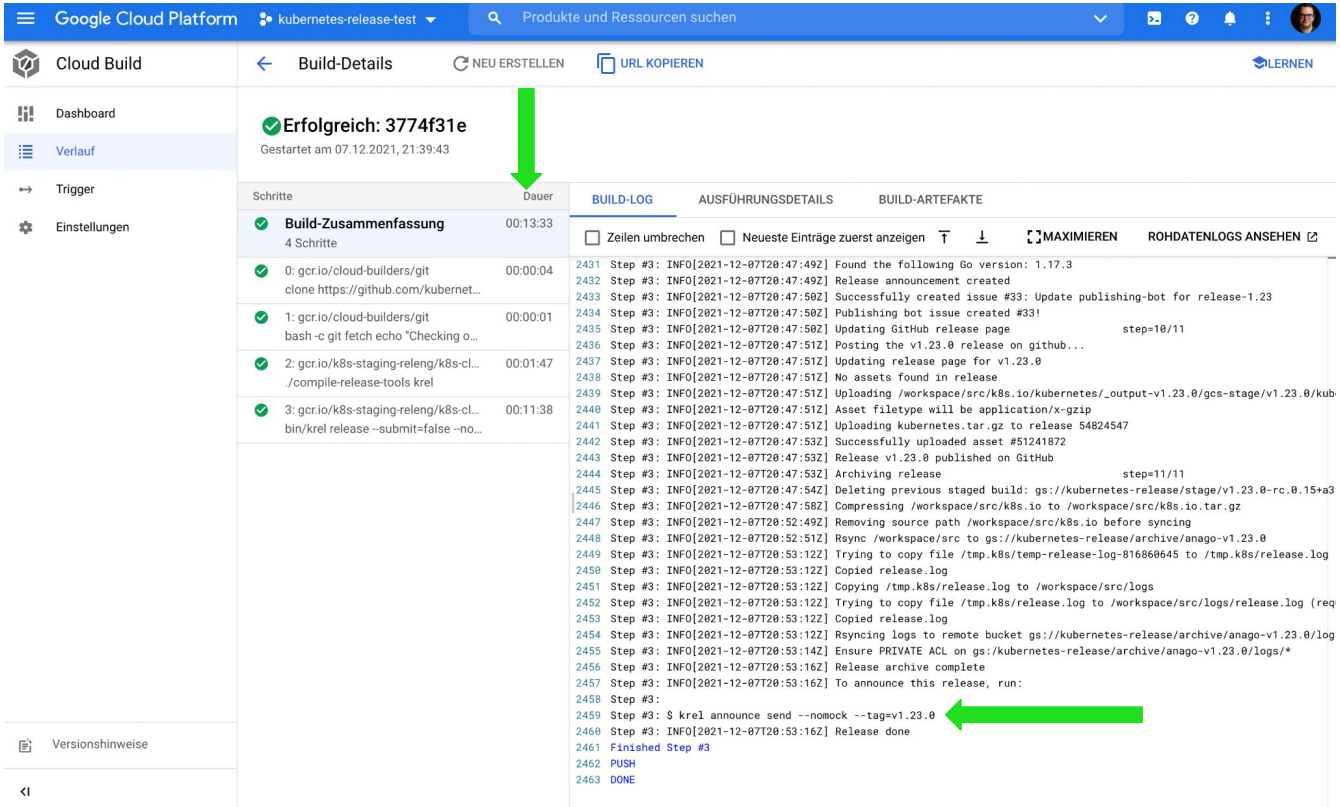


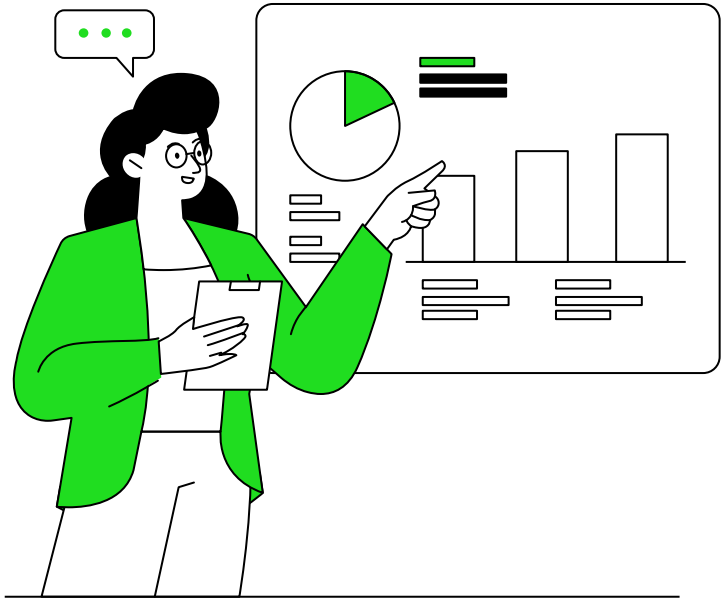
# YOU NEED TO KNOW WHAT YOU ARE LOOKING FOR



# EVERY BUILD RUNS IN THE CLOUD





# LESSONS LEARNED

06

# TAKE AWAY

## STRONG AUTOMATION

Every single step within the release is automated and tested

## WITHOUT A RELEASE TEAM IT WILL NOT WORK

Even with a high automation you need someone to ensure the quality and drive further development

## TESTS MUST BE CONSISTENT

Sometime a commit is just a few lines long, sometimes they are tens of thousands of lines

## NEVER TRUST A COMMIT

“Everyone” can commit, every commit & PR is tested and reviewed

## OWN PROCESSES

… require sometimes own tools

## DID I MENTION ATOMIZATION?

# THANK YOU!

Max Körbächer - Co-Founder