How a modern Yocto setup could look like

Anna-Lena Marx, inovex

Yocto Project Summit, 2022.11

Anna-Lena Marx

@Allegra3141

Allegra
@social.linux.pizza

Allegra42

Embedded Systems Dev at inovex

- Linux Kernel
- Yocto
- Android Embedded

B.Sc. Computer Science

M.Sc. Embedded Systems

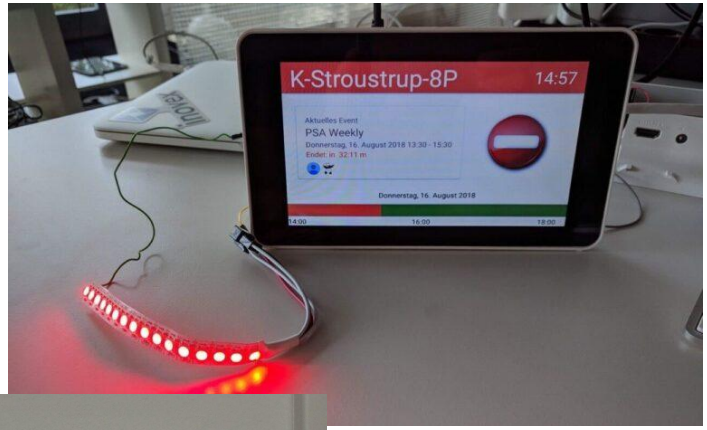
B.Eng. Electrical Engineering - *ongoing Hobby*

inovex

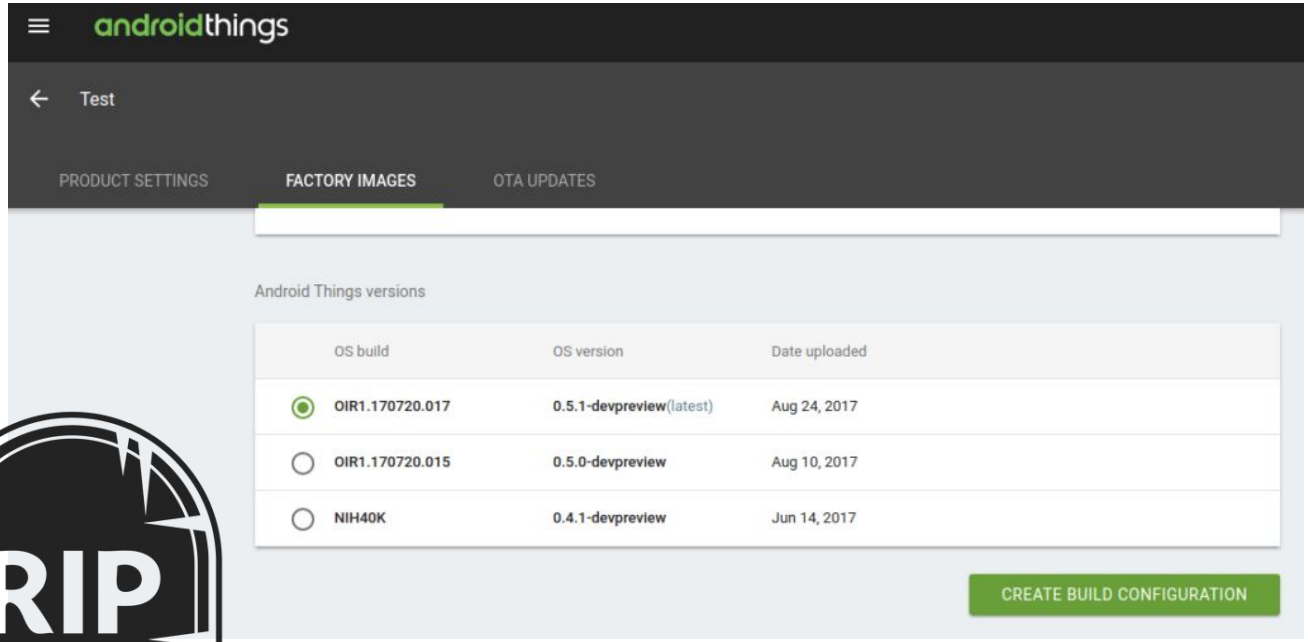
The Story behind
Meeting Room Information Screens at inovex

~ 25 Meeting rooms spread on 5 cities

before Corona -> lots of Meetings



inovex



Android Things was shutdown in 2022

[Killed by Google: Google Graveyard]

inovex

Reboot

What do we wish for a modern, maintainable System?

Our wishlist

- ❏ Full-stack patchability
- ❏ Version control
- ❏ Reproducible builds
- ❏ Long-Term maintainable
- ❏ Proper license (and version) management
- ❏ Android-like, secure and stable update mechanism
- ❏ Release management Over-the-Air
- ❏ Continuous Integration
- ❏ A vendor independent system on all levels

- ❑ **Full stack patchability**
- ❑ **Version control**
- ❑ **Reproducible builds**
- ❑ **Long-Term maintainable**

✓ just use yocto PROJECT

→ Hardware can be reused, no invest needed
→ all points above are fulfilled by design

Managing Yocto - a side note

Name	Last commit
..	
meta-inovex	rename inovex-media => inovexmedia
meta-mender @ 5b518cc7	update layers
meta-openembedded @ ab9fca48	update layers
meta-raspberrypi @ 934064a0	Update submodule meta-raspberrypi in oder to fix linux-...
meta-security @ b76698c7	update layers
meta-virtualization @ c5f61e54	update layers
poky @ bba32338	update layers

```
<?xml version='1.0' encoding='UTF-8'?>
<manifest>
  <phytec pdn="PD21.1.0" release_uid="BSP-Yocto-FSL-i.MX8M-PD21.1.0" soc="iMX8M" supported_builds="
    phyboard-polaris-imx8m-3/phytec-headless-bundle/yogurt-vendor,
    phyboard-polaris-imx8m-3/phytec-headless-image/yogurt-vendor-secure,
    phyboard-polaris-imx8m-3/phytec-qt5demo-image/yogurt-vendor-xwayland,
    phyboard-polaris-imx8m-3/phytec-vision-image/yogurt-vendor-xwayland,
    phyboard-polaris-imx8m-4/-c populate_sdk phytec-qt5demo-image/yogurt-vendor-xwayland,
    phyboard-polaris-imx8m-4/phytec-headless-bundle/yogurt-vendor,
    phyboard-polaris-imx8m-4/phytec-headless-image/yogurt-vendor-secure,
    phyboard-polaris-imx8m-4/phytec-qt5demo-image/yogurt-vendor-xwayland,
    phyboard-polaris-imx8m-4/phytec-vision-image/yogurt-vendor-xwayland
  " bspextension="FSL" />

  <default revision="zeus" sync-j="2" remote="git.phytec" />

  <remote fetch="https://git.yoctoproject.org/git" name="yocto" />
  <remote fetch="https://github.com/Freescale" name="community" />
  <remote fetch="https://github.com/openembedded" name="oe" />
  <remote fetch="https://github.com/OSSystems" name="OSSystems" />
  <remote fetch="https://github.com/meta-qt5" name="QT5" />
  <remote fetch="https://github.com/meta-rust" name="rust" />
  <remote fetch="git://git.openembedded.org" name="python2" />
  <remote fetch="https://source.codeaurora.org/external/imx" name="CAF" />
  <remote fetch="https://github.com/rauc" name="rauc" />
  <remote fetch="https://github.com/kraj" name="clang" />
  <remote name="git.phytec" fetch="git://git.phytec.de" />
  <remote name="ssh.phytec" fetch="ssh://git@git.phytec.de" />

  <project name="poky" path="sources/poky" remote="yocto" revision="d88d62c20d7d8da85f02edb170dae0280624ad7e">
    <ignorebaselayer />
    <sublayer path="meta" />
    <sublayer path="meta-poky" />
  </project>

  <project name="meta-openembedded" path="sources/meta-openembedded" remote="oe" revision="2b5dd1eb81cd08bc065bc76125f2856e93
    <ignorebaselayer />
    <sublayer path="meta-oe" />
    <sublayer path="meta-networking" />
    <sublayer path="meta-python" />
    <sublayer path="meta-multimedia" />
    <sublayer path="meta-filesystems" />
    <sublayer path="meta-perl" />
    <sublayer path="meta-gnome" />
  </project>
```

inovex

Managing Yocto - a side note

There are lots of valid ways to work with Yocto - We moved to KAS because

- easy readable, clear syntax
- easy to use for non Yocto people
- comes already with a container!
  - really nice for CI
- persistent way to edit local.conf

✓ **handle Yocto easier**

```yaml
header:
  version: 11

distro: poky

defaults:
  repos:
    refspec: master

repos:
  poky:
    url: https://git.yoctoproject.org/git/poky
    path: "layers/poky"

    refspec: kirkstone
    layers:
      meta:
      meta-poky:
      meta-yocto-bsp:

  meta-openembedded:
    url: https://git.openembedded.org/meta-openembedded
    path: "layers/meta-openembedded"
    refspec: kirkstone
    layers:
      meta-oe:
      meta-python:
      meta-filesystems:
      meta-networking:

  meta-mender:
    url: https://github.com/mendersoftware/meta-mender.git
    path: "layers/meta-mender"
    refspec: kirkstone
    layers:
      meta-mender-core:

  meta-virtualization:
    url: https://git.yoctoproject.org/meta-virtualization
    path: "layers/meta-virtualization"
    refspec: kirkstone

  meta-inovex:
    url: ssh://git@gitlab.inovex.de:2424/inovex-raumplanung/meta-inovex.git
    path: "layers/meta-inovex"
    refspec: master


local_conf_header:
  roombooking-base: |
    PACKAGE_CLASSES = "package_deb"
```

inovex

❑ Proper license (and version) management

Why?

- we want/need to be license compliant
- we want to know exactly **what** we ship in our image
  - Component Name
  - Version
  - License

= Software Bill of Materials (SBOM)

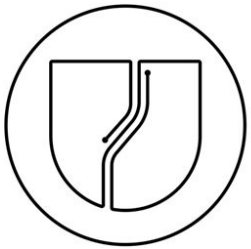
✓ **INHERIT += "create-spdx"**

- **Android like, secure and stable update mechanism**
- **Release management Over-the-Air**

More exactly

- Image based updates
- A/B updates as state of the art
  - with a rollback/recovery mechanism
- A matching server implementation
  - allows starting updates for different groups of devices
  - OTA
  - a bit like Android Things console

- ❑ Android like, secure and stable update mechanism
- ❑ Release management Over-the-Air

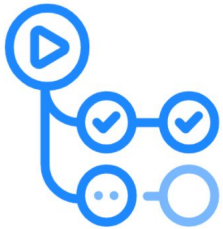


✓ and our winner is …

… Mender

inovex

❑ Continuous Integration



GitHub Actions

CI/CD

Jenkins

✓ we go with … GitLab

inovex

And what's about Continuous Delivery or Deployment?

- we do **not** want completely automated deployment on embedded devices!

- Yocto with CI gives us deployable artifacts
- Mender make rolling out easy, nevertheless

❑ A vendor independent system on all levels

- hardware can be changed or operated in parallel
- various options on software-side
  - app
  - update system
  - CI/CD
  - we could even switch to Buildroot …

✓ **switching a component is not effortless,**
but possible without dropping the whole stack

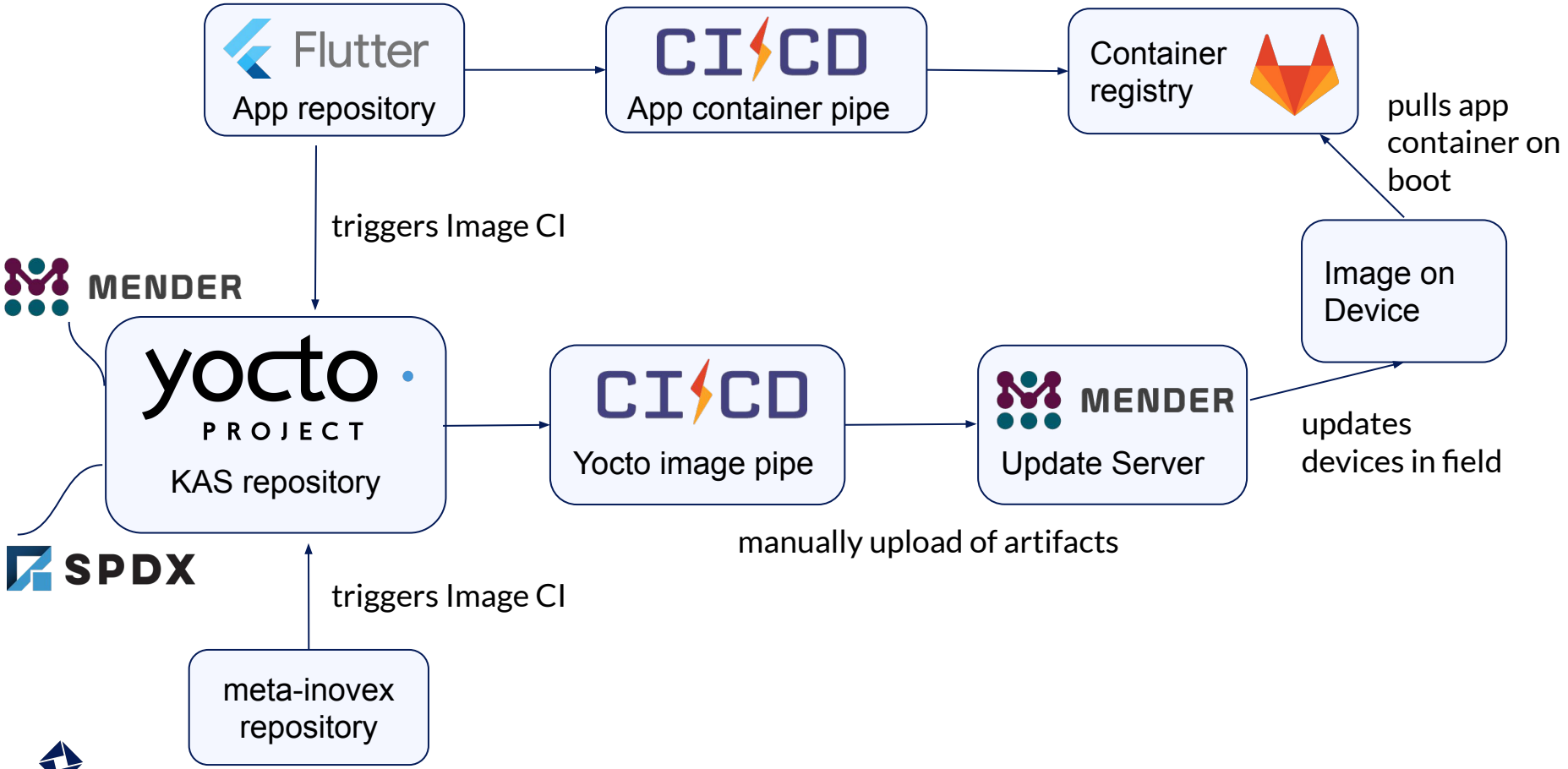
The App side

Flutter

- flutter-pi Embedder
  - UI renders directly on GPU
  - no X11 / Wayland, … needed
- no JavaScript / Webbrowser mess in Yocto
- keeps the complexity of Yocto away from app devs
- ships in a container for easy changes and updates

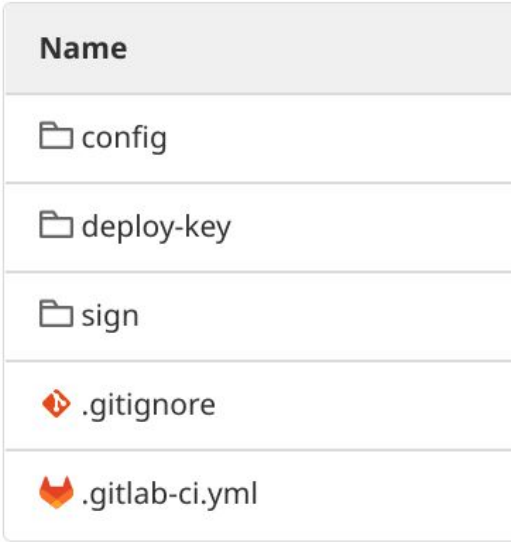
✓ **nice application without major pain points**

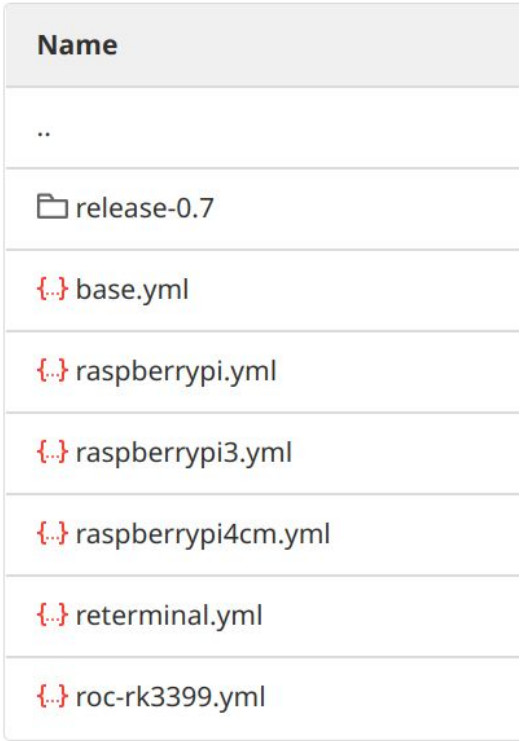
How it looks today


Flutter
App repository
CI/CD
App container pipe
Container registry
pulls app container on boot
triggers Image CI
MENDER
Image on Device
yocto PROJECT
KAS repository
CI/CD
Yocto image pipe
MENDER
Update Server
updates devices in field
manually upload of artifacts
SPDX
triggers Image CI
meta-inovex repository

inovex

The KAS config repository





Yocto CI

```
35    script:
36        - echo "Starting KAS Yocto build"
37        - kas checkout config/raspberrypi3.yml
38        - sed -i "s#{{ DOCKER_REGISTRY_AUTH }}#${DOCKER_REGISTRY_AUTH}#" layers/meta-inovex/recipes-config/inovex-config/files/dock
39        - sed -i "s#kirkstone-dev#kirkstone-dev-${CI_JOB_STARTED_AT}#" config/base.yml     # add date to dev builds
40        - kas build config/raspberrypi3.yml
41        - echo "Yocto image for Raspberry Pi 3 finished."
```

Trigger CI when changing meta-inovex

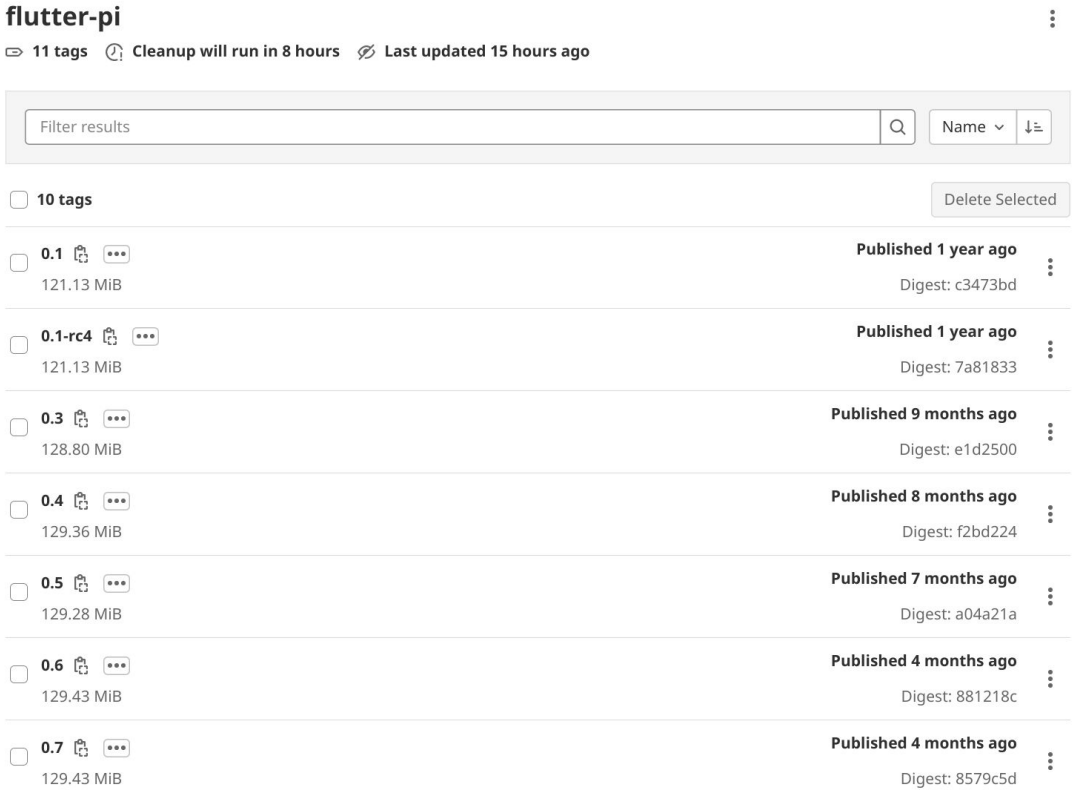
.gitlab-ci.yml 424 bytes Edit

```
1  stages:          # List of stages for jobs, and their order of execution
2    - trigger-build
3
4  trigger-build:
5    image: ubuntu:impish
6    tags:
7      - shared
8    stage: trigger-build
9    before_script:
10     - apt-get update && apt-get install -y curl
11   script:
12     - echo "Trigger KAS Yocto CI"
13     - curl -X POST --fail -F token=[redacted] -F ref=main https:
14
```

[Doc: Trigger pipelines in GitLab]

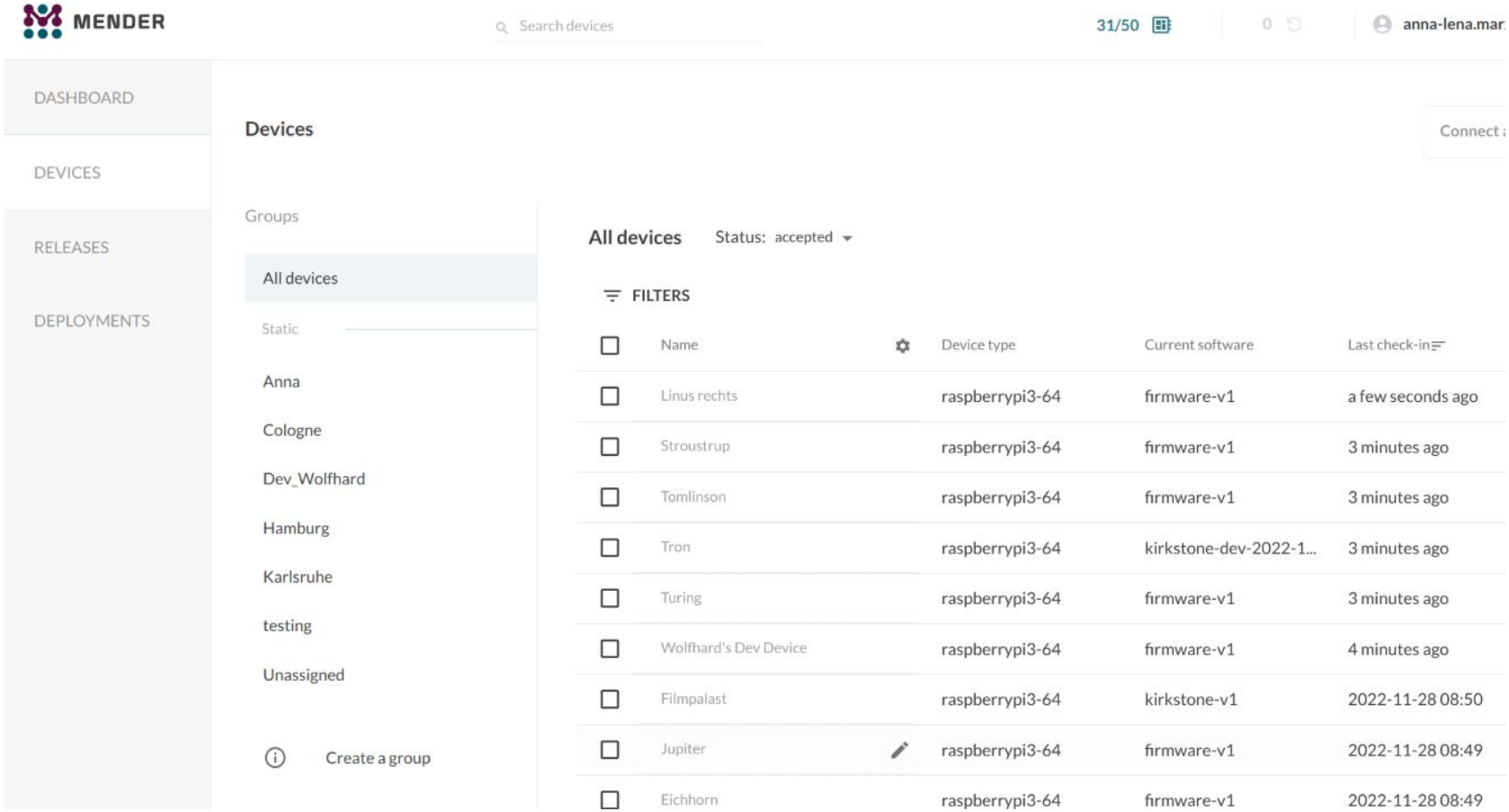
inovex

Container Registry

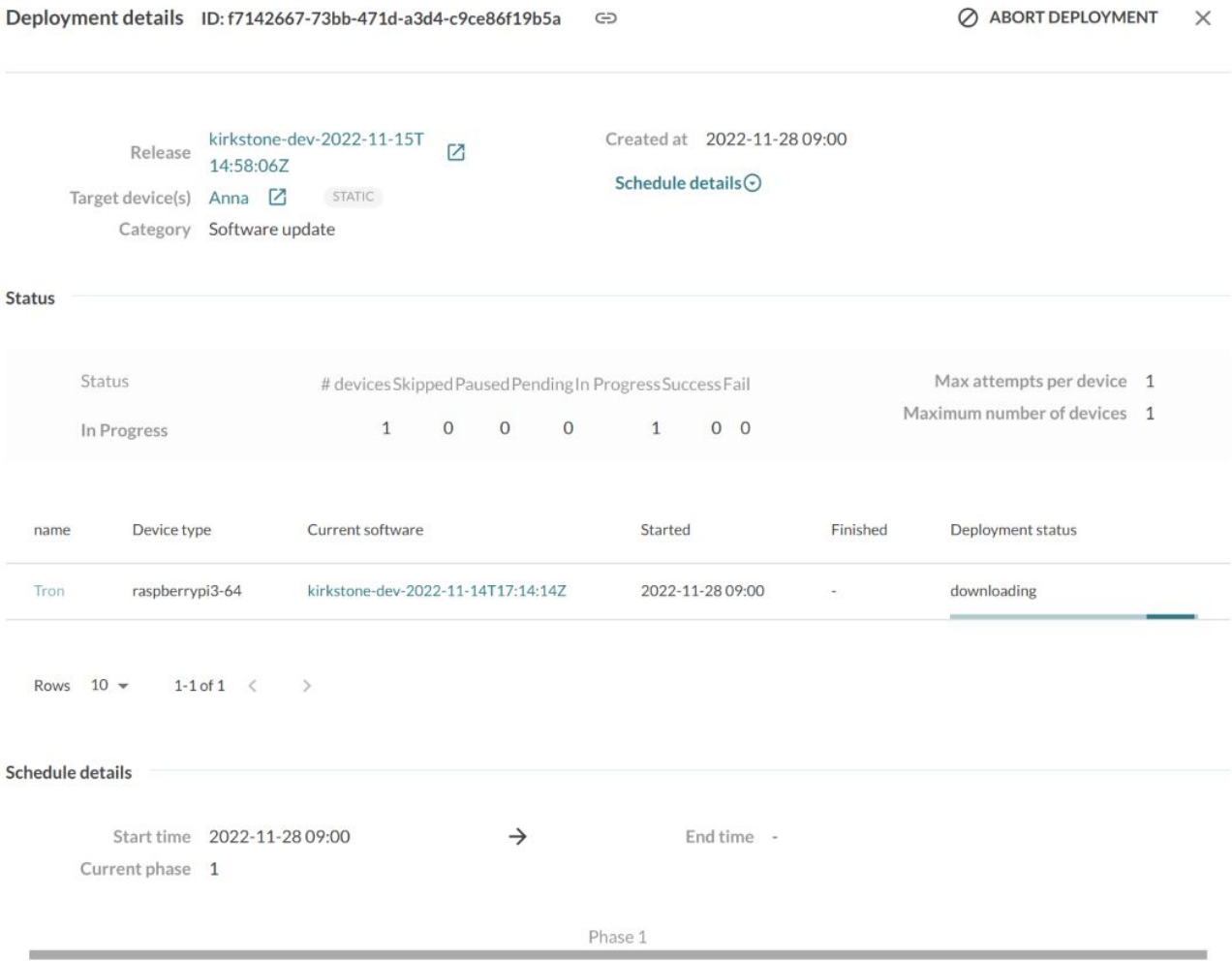


inovex

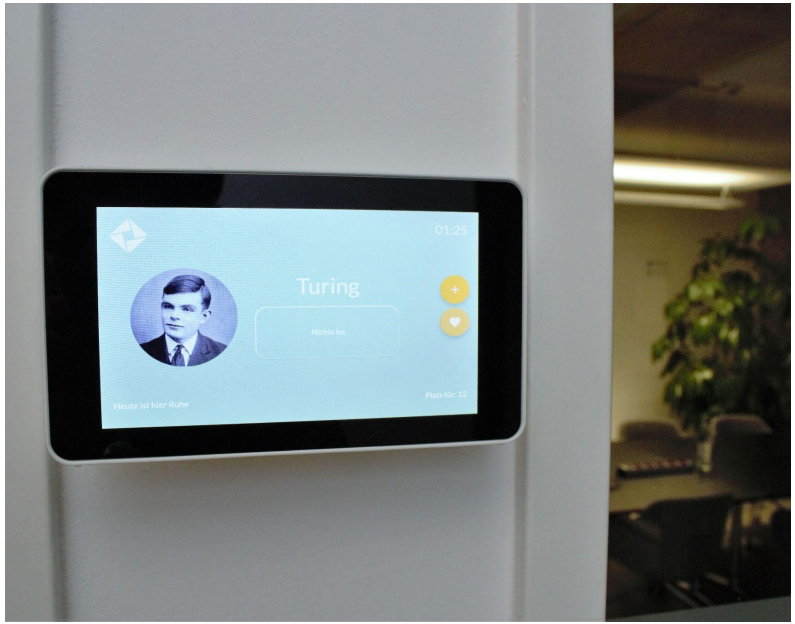
Mender

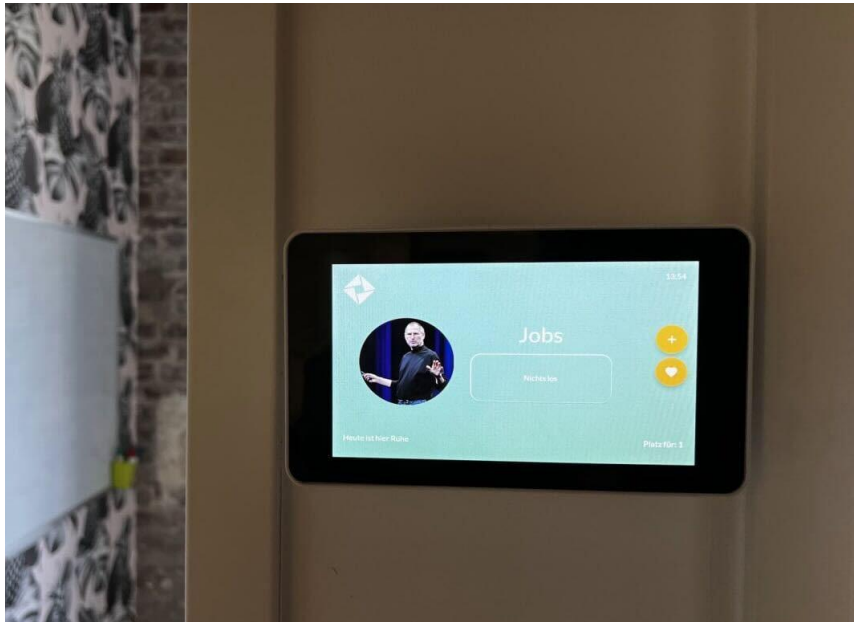


inovex


Deployment details ID: f7142667-73bb-471d-a3d4-c9ce86f19b5a
ABORT DEPLOYMENT
Release kirkstone-dev-2022-11-15T 14:58:06Z
Target device(s) Anna STATIC
Category Software update
Created at 2022-11-28 09:00
Schedule details
Status
Status # devices Skipped Paused Pending In Progress Success Fail
In Progress 1 0 0 0 1 0 0
Max attempts per device 1
Maximum number of devices 1
name Device type Current software Started Finished Deployment status
Tron raspberrypi3-64 kirkstone-dev-2022-11-14T17:14:14Z 2022-11-28 09:00 - downloading
Rows 10 1-1 of 1
Schedule details
Start time 2022-11-28 09:00
End time -
Current phase 1
Phase 1

The result





inovex

Learnings and recommendations

What we learned

- Be aware that Google burry a lot of projects
- Have a backup plan if one of your components becomes obsolete or is not longer maintained
- Involve internal expertise
  - use an internal project to test new things before using them in real world

Building and maintaining an embedded device is a huge amount of work and involves a wide range of expertise.

Find us, follow us

@inovexgmbh

@inovexlife

inovex

Thank you!

Anna-Lena Marx
anna-lena.marx@inovex.de

yocto PROJECT | THE LINUX FOUNDATION

KAS configs

{..} base.yml 3.60 KiB

```yaml
header:
  version: 11

distro: poky

defaults:
  repos:
    refspec: master

repos:
  poky:
    url: https://git.yoctoproject.org/git/poky
    path: "layers/poky"

    refspec: kirkstone
    layers:
      meta:
      meta-poky:
      meta-yocto-bsp:

  meta-openembedded:
    url: https://git.openembedded.org/meta-openembedded
    path: "layers/meta-openembedded"
    refspec: kirkstone
    layers:
      meta-oe:
      meta-python:
      meta-filesystems:
      meta-networking:

  meta-mender:
    url: https://github.com/mendersoftware/meta-mender.git
    path: "layers/meta-mender"
    refspec: kirkstone
    layers:
      meta-mender-core:

  meta-virtualization:
    url: https://git.yoctoproject.org/meta-virtualization
    path: "layers/meta-virtualization"
    refspec: kirkstone

  meta-inovex:
    url: ssh://git@gitlab.inovex.de:2424/inovex-raumplanung/meta-inovex.git
    path: "layers/meta-inovex"
    refspec: master


local_conf_header:
  roombooking-base: |
    PACKAGE_CLASSES = "package_deb"
    EXTRA_IMAGE_FEATURES = "debug-tweaks"
    INIT_MANAGER = "systemd"
    DISTRO_FEATURES:append = " wifi"
    DISTRO_FEATURES:append = " virtualization"
    IMAGE_INSTALL:append = " iw wpa-supplicant packagegroup-base"
    IMAGE_INSTALL:append = " docker-ce cgroup-lite ca-certificates util-linux kernel-modules"
    IMAGE_INSTALL:append = " ca-certificates util-linux kernel-modules"
    IMAGE_INSTALL:append = " jq"
    IMAGE_INSTALL:append = " inovex-config"
    IMAGE_INSTALL:append = " psplash-inovex"
    CMDLINE:append = " quiet"
    DISABLE_SPLASH = "1"
    IMAGE_FSTYPES:remove = " wic wic.bz2 wic.bmap "

    # enable SPDX SBOM generation
    INHERIT += "create-spdx"
    SPDX_PRETTY = "1"

    SPDX_SUPPLIER = "Organization: inovex GmbH"
    SPDX_NAMESPACE_PREFIX = "http://inovex.de/spdx"
    SPDX_UUID_NAMESPACE = "inovex.de"

  mender-base: |
    CONF_VERSION = "2"
    INHERIT += "mender-full"
    INIT_MANAGER = "systemd"

    IMAGE_INSTALL:append = " mender-connect"

    # how often to check for available firmware updates
    MENDER_UPDATE_POLL_INTERVAL_SECONDS = "60"
    # how often to report inventory status
    MENDER_INVENTORY_POLL_INTERVAL_SECONDS = "600"
```

inovex

The final stack

CI/CD

Flutter

MENDER

yocto PROJECT

SPDX

Thank you!

Anna-Lena Marx
Embedded Systems Dev

inovex GmbH
Ludwig-Erhard-Allee 6
76131 Karlsruhe

anna-lena.marx@inovex.de

inovex

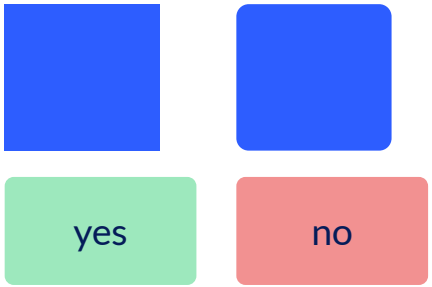
Farbe & Gestaltung

- Unsere Grundfarbe ist #051c59
- Die Grundfarbe wird mit einer Akzentfarbe kombiniert (hier #7df481)
- Titel stehen in Lato Bold 28pt
- Text steht in Lato Regular 18–24pt
- Zusätzliche Akzentfarben sind im Layout hinterlegt
- Font Check: Das hier ist Lato!

Das hier ist Lato!

It’s dangerous to go alone, take this!

- Eine Liste mit korrekten Aufzählungszeichen
- Punkt 2
  - Einrückung

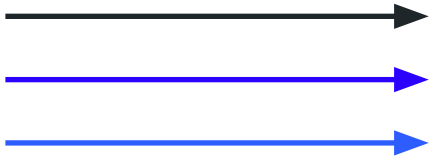


Eine Karte

Mit Überschrift und Textfeld

Form einfach

Form gestylt



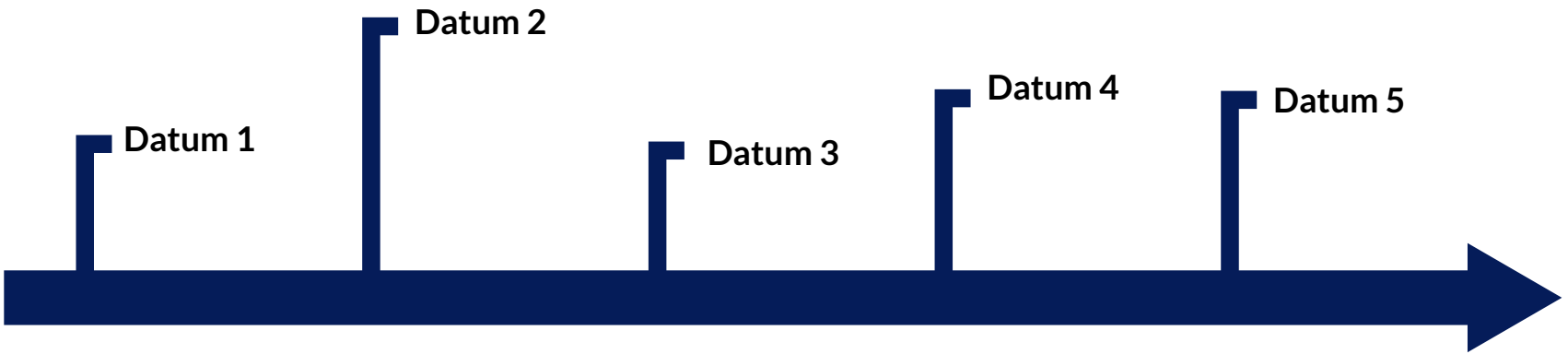
Pfeile ohne Schatten

So kann Code formatiert werden

```
if you.dontlike(pseudocode)
   try
      use.(realcode)
   catch
      me
         if you.can()

   return “to Sender”
```

Time flies like an arrow



inovex