Inhaltsverzeichnis

1 Diskussion: Dev-Sec-Ops

1.1 Warum Dev-Sec-Ops

1.2 Für wen ist Dev-Sec-Ops?

2 Der „ideale“ Dev-Sec-Ops Prozess

2.1 Welche Werkzeuge kommen zum Einsatz?

2.2 Wie ordnet man seine Pipeline am besten an?

3 Dev-Sec-Ops Werkzeug: License Checker

3.1 Auswahl des geeignetsten Werkzeugs

3.2 Inbetriebnahme

4 Literaturverzeichnis

[1] D. Glass, *NPM License Checker*, https://www.npmjs.com/package/license-checker (besucht am 12. 09. 2021).

[2] D. Bauernfeind, *Composer License Checker*, https://github.com/dominikb/composer-license-checker (besucht am 12. 09. 2021).

5 Anhänge

.gitlab-ci.yml

```yaml
check-npm-licenses:
  image: node:16
  script:
    - /bin/bash ./license-checker/npm-licenses.sh
      ↪ "MIT;ISC;Apache-2.0;0BSD;BSD-2-Clause;BSD-3-Clause;CC0-1.
      ↪ 0;CC-BY-4.0;Unlicense"
  artifacts:
    when: always
    paths:
      - npm_license_summary.txt
      - npm_license_detailed.json
    expire_in: 3 days
    reports:
      junit:
        - npm_licenses.xml

check-composer-licenses:
  image: composer:2.1.6
  script:
    - /bin/bash ./license-checker/composer-licenses.sh
      ↪ "MIT;LGPL-2.1;BSD-3-Clause;LGPL-3.0;Apache-2.0"
  artifacts:
    when: always
    paths:
      - composer_license_summary.txt
      - composer_license_detailed.txt
    expire_in: 3 days
    reports:
      junit:
        - composer_licenses.xml
```

license-checker/npm-licenses.sh

```
# install the npm dependency
npm install
npm install -g license-checker
npm install -g yui-lint

# list the summary of licenses used
license-checker --summary --out "npm_license_summary.txt"
# generate a overview of licenses
license-checker --json --out "npm_license_detailed.json"


fails=0
# Generate Output.xml
echo '<?xml version="1.0" encoding="UTF-8"?><testsuites><testsuite
 ↪  id="NPM-CUSTOM-LICENSE-CHECKER">' > npm_licenses.xml
echo '<testcase name="NPM check allowed Licenses">' >>
 ↪  npm_licenses.xml

# execute test and capture error stream in variable
ERROR=$(license-checker --onlyAllow "$1" 2>&1 >/dev/null)

if [ $? != 0 ];
then
    let "fails++"
    echo '<failure type="FAILURE">' >> npm_licenses.xml
    echo $ERROR >> npm_licenses.xml
    echo '</failure>' >> npm_licenses.xml
else
    echo '<passed type="PASSED"></passed>' >> npm_licenses.xml
fi

echo '</testcase>' >> npm_licenses.xml
echo '</testsuite></testsuites>' >> npm_licenses.xml

# 0 if all succeeded, larger 0 otherwise
exit $fails
```

license-checker/composer-licenses.sh

```
# install the composer dependency
composer require dominikb/composer-license-checker

# list the summary of licenses used
./vendor/bin/composer-license-checker report > "composer_license_summary.txt"

# generate a overview of licenses
# as there's no real option for this, run a check that lists everything and remove the Error-messages
./vendor/bin/composer-license-checker check --allowlist NAMEOFNOLICENSE >  "composer_license_detailed.txt"
sed -i '/\[ERROR\]/d' composer_license_detailed.txt

# format the first operand correctly to be used in the command
OPTIONS=""
IFS=';' read -ra ADDR <<< "$1"
for i in "${ADDR[@]}"; do
  OPTIONS="$OPTIONS --allowlist $i"
done


fails=0
# Generate Output.xml
echo '<?xml version="1.0" encoding="UTF-8"?><testsuites><testsuite id="COMPOSER-CUSTOM-LICENSE-CHECKER">' > composer_licenses.xml
echo '<testcase name="Composer check allowed Licenses">' >> composer_licenses.xml

# execute test and capture output stream in variable
OUTPUT=$(./vendor/bin/composer-license-checker check $OPTIONS)

if [ $? != 0 ];
then
    let "fails++"
    echo '<failure type="FAILURE">' >> composer_licenses.xml
    echo $OUTPUT >> composer_licenses.xml
    echo '</failure>' >> composer_licenses.xml
else
```

```
35      echo '<passed type="PASSED"></passed>' >> composer_licenses.xml
36  fi
37
38  echo '</testcase>' >> composer_licenses.xml
39  echo '</testsuite></testsuites>' >> composer_licenses.xml
40
41  # 0 if all succeeded, larger 0 otherwise
42  exit $fails
```