Wildfly, Quarkus og brugen af specifikationer (Java EE 8)

Hvad skal jeg tale om

Wildfly . . . 1
Microprofile . . . 2
Quarkus . . . 2
- Extensions . . . 3
- Maven . . . 3

ToDo gennemgang . . . 4
- MicroProfile Config . . . 4
- JAX-RS . . . 4
- Persistering af data . . . 4
- Sikkerhed . . . 5
- MicroProfile Health . . . 6
- MicroProfile Metrics . . . 6
- OpenAPI and SwaggerUI . . . 7

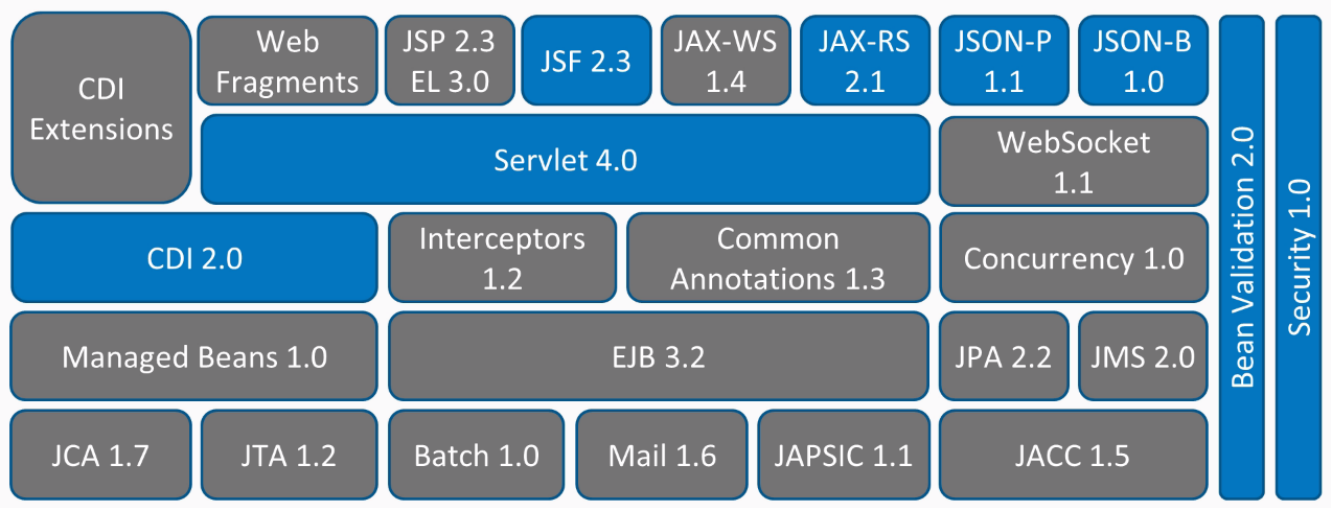
Bookstore gennemgang . . . 7
- Author JAX-RX + service . . . 7
- Book JAX-RX + extends PanacheEntity . . . 7
- Publisher RestController/RequestMapping (Spring) + service . . . 7
- Test @QuarkusTest . . . 7

Appendix . . . 7

Wildfly

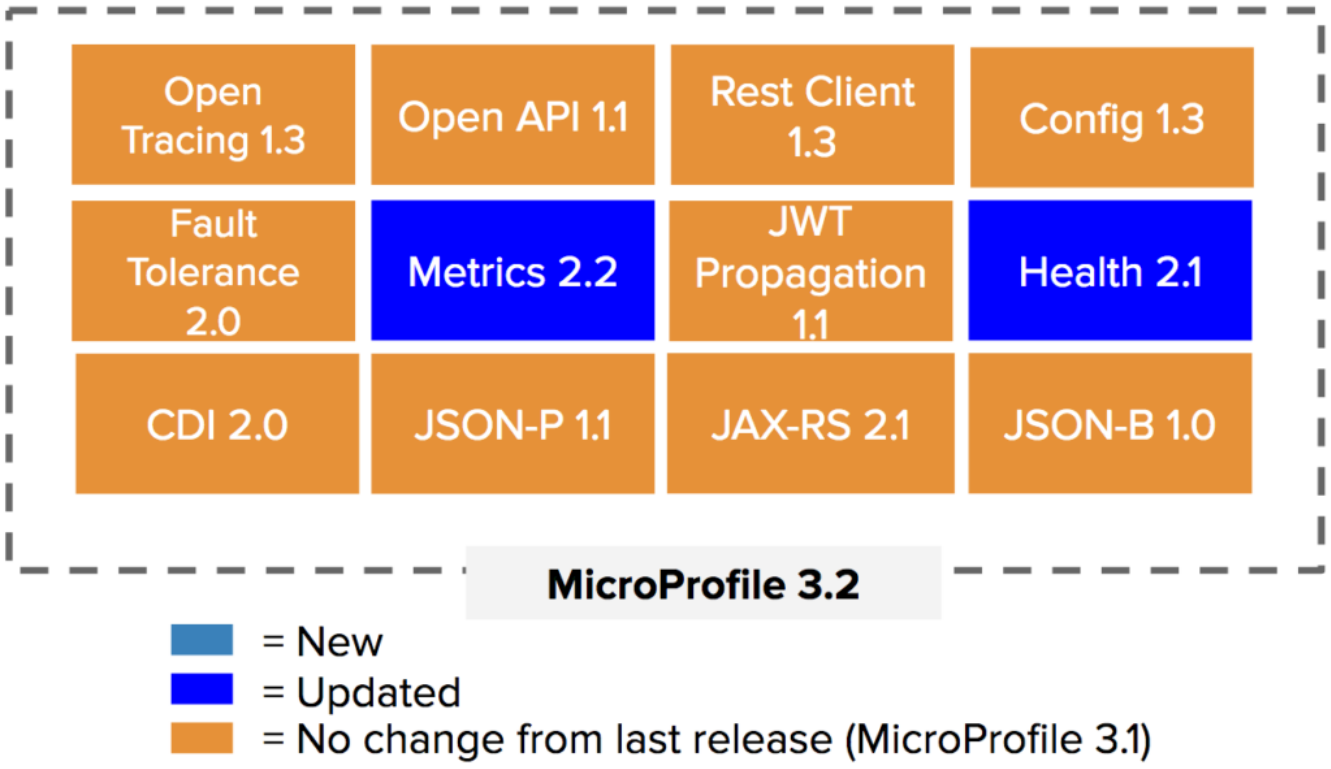
Wildfly

- JBoss EAP 7.1 (Wildfly 14)
- Java EE 8
- Jakarta EE 8
- Moduler
- beans.xml



Microprofile

Microprofile

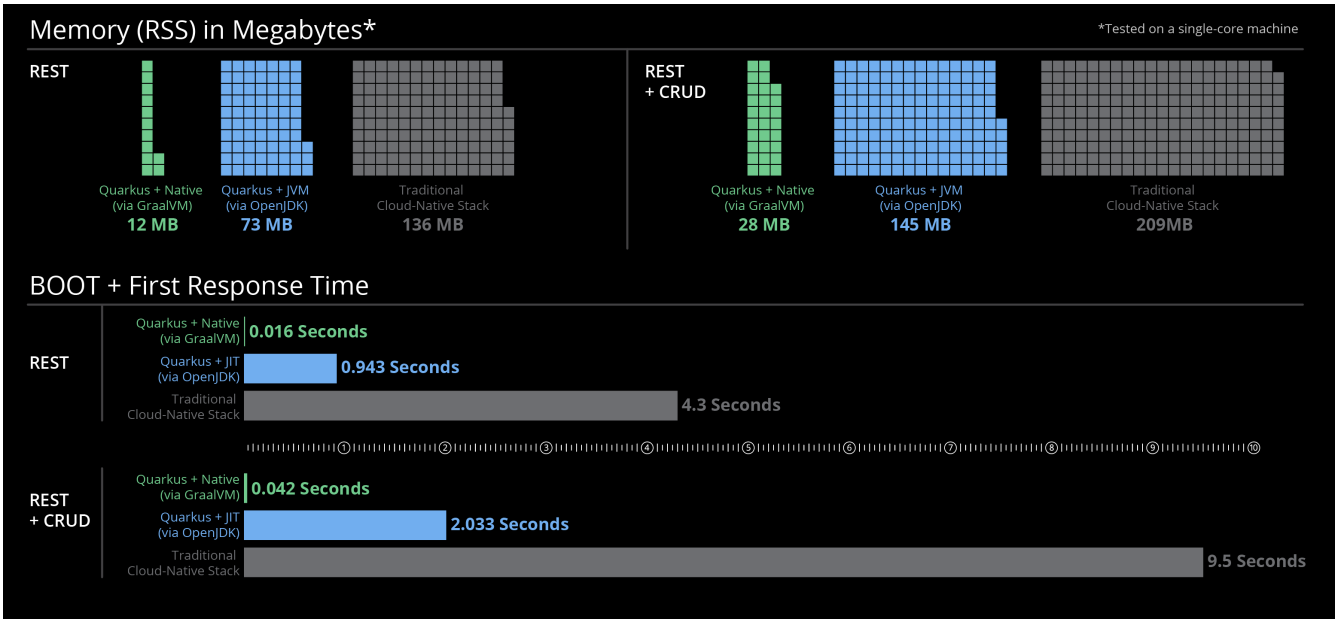


Eclipse MicroProfile 3.2 is Now Available

Quarkus

Supersonic Subatomic Java

A Kubernetes Native Java stack tailored for OpenJDK HotSpot and GraalVM, crafted from the best of breed Java libraries and standards.



Quarkus

GraalVM

Dependency Injection

- Quarkus - Contexts and Dependency Injection
- Weld
- beans.xml vs. annotations

Extensions

- Sammenligning med Wildflys moduler
- Quarkus - Start coding with code.quarkus.io

Maven

```
./mvnw compile quarkus:dev
```

```
./mvnw clean package
java -jar target/quarkus-fagligfredag-demo-0.0.1-SNAPSHOT-runner.jar
```

```
./mvnw package -Pnative
./target/quarkus-fagligfredag-demo-0.0.1-SNAPSHOT-runner
```

ToDo gennemgang

Dette bliver en introduktion til Quarkus med udgangspunkt i en simpel ToDo applikation.

Vi ser på hvordan kode skrevet til Java EE 8 specifikation, kan laves om til Quarkus.

Vi vil se på disse punkter i gennemgangen

- Konfiguration af applikationen
- Rest service
- Persistering af data i database (PostgreSQL) med JPA (Hibernate)
- Tilføj sikkerhed til applikationen
- Tilføj Health
- Tilføj Metrics
- Tilføj OpenAPI (swagger)

Applikationen afvikler vi med Quarkus og Wildfly.

Koden vil blive startet fra

- Terminal
- Docker
- Kubernetes (minikube)

MicroProfile Config

PingResource

```
@ConfigProperty(name = "pingMessage", defaultValue = "pingMessage need config..")
```

- ENV
- Propeties

JAX-RS

```
@Path("todos")
@Consumes(MediaType.APPLICATION_JSON)
@Produces(MediaType.APPLICATION_JSON)
```

Persistering af data

Hibernate

```
@Inject
EntityManager entityManager;
```

Wildfly - persistence.xml

```
<persistence xmlns="http://xmlns.jcp.org/xml/ns/persistence"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence
    http://xmlns.jcp.org/xml/ns/persistence/persistence_2_1.xsd"
    version="2.1">

    <persistence-unit name="ToDoPU" transaction-type="JTA">
        <description>My ToDo entities</description>
        <jta-data-source>jboss/datasources/ToDoDS</jta-data-source>
        <properties>
            <property name="hibernate.dialect"
                value="org.hibernate.dialect.PostgreSQL95Dialect" />
            <property name="hibernate.show_sql"
                value="false" />
            <property name="hibernate.format_sql"
                value="true" />
            <property name="javax.persistence.schema-generation.database.action"
                value="drop-and-create" />
            <property name="javax.persistence.validation.mode"
                value="NONE" />
        </properties>
    </persistence-unit>

</persistence>
```

Quarkus - application.properties

```
quarkus.datasource.url=jdbc:postgresql://PostgreSQLDemo:5432/hibernate_db
quarkus.datasource.driver=org.postgresql.Driver
quarkus.datasource.username=hibernate
quarkus.datasource.password=hibernate
quarkus.datasource.max-size=8
quarkus.datasource.min-size=2
quarkus.hibernate-orm.database.generation=drop-and-create
quarkus.hibernate-orm.log.sql=false
```

Sikkerhed

```
@Inject
Principal principal;
```

```
@Inject
JsonWebToken jwt;
```

```
export
TOKEN="eyJraWQiOiIvcHJpdmF0ZUtleS5wZW0iLCJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJpc3MiOi
JodHRwczovL3F1YXJrdXMuaW8vdXNpbmctand0LXJiYWMiLCJqdGkiOiJhLTEyMyIsInN1YiI6Im1pY2JuLXVz
aW5nLWp3dC1yYmFjIiwidXBuIjoibWljYm5AcmVkcGlsbC1saW5wcm8uY29tIiwicHJlZmVycmVkX3VzZXJuYW
1lIjoibWljYm4iLCJhdWQiOiJ1c2luZy1qd3QtcmJhYyIsImJpcnRoZGF0ZSI6IjIwMDEtMDctMTMiLCJyb2xl
TWFwcGluZ3MiOnsiZ3JvdXAxIjoiR3JvdXAxTWFwcGVkUm9sZSIsImdyb3VwMiI6Ikdyb3VwMk1hcHBlZFJvbG
UifSwiZ3JvdXBzIjpbIkVjaG9lciIsIlRlc3RlciIsIlN1YnNjcmliZXIiLCJncm91cDIiLCJ1c2VyIl0sImlh
dCI6MTU3NjM0MTk2OSwiYXV0aF90aW1lIjoiTnVtZXJpY0RhdGV7MTU3NjM0MTk2OSAtPiAxNC1EZWMtMjAxOS
AxNzo0NjowOSBDRVR9IiwiZXhwIjoxNTc2MzQyMjY5fQ.FYGtGIT_iU2hEF2F_XvjMXyCzCD7_Q4UXpNwhRJhp
bRTqhFOeqQ-1yJllr6xs1oXH7OdRNovmqXjVZp3NEARwF7we-
bm_h1TFPOQECeHGl8jvE7Ig9Cx0wL0QDxCUQKLQ1pOa9RSwy-_BkkJ0d3Nukn-
Q871a6wm34syIWYB10KWWnm9CBRpeTRkA9YO-
_xNhFqkmeDSkUfdHvoRbXbHEgg_TtB3_IOJp1xtC5xmrytm0Q-
CmwdxnpHwAPfgVCJLPqg2bI_20_ESkKvLeMkFWTV3ReHSr4GNkEWOAcmCCajP9kizvftoPO2WyB0AkcdgzPLgW
4_5D6Sd3L64ER8S9Q"

curl -v http://localhost:8080/todos \
    -H 'Accept: application/json' \
    -H 'Authorization: Bearer '$TOKEN'' \
    -H 'Content-Type: application/json' \
    -d '{"subject":"Hello from Quarkus","body":"Content","priority": 1,"importens":
10,"owner" : "Duke"}'
```

MicroProfile Health

```
@Health
```

```
curl -X GET \
    http://localhost:8080/health \
    -H 'Accept: application/json'
```

quarkus.io - Health Guide

MicroProfile Metrics

```
curl -X GET \
    http://localhost:8080/metrics/application \
    -H 'Accept: application/json'
```

quarkus.io - Metrics Guide

OpenAPI and SwaggerUI

Just add `quarkus-smallrye-openapi` as a dependency in `pom.xml` and Bob is your uncle.

```
<dependency>
    <groupId>io.quarkus</groupId>
    <artifactId>quarkus-smallrye-openapi</artifactId>
</dependency>
```

- OpenAPI
- swagger-ui

OpenAPI and Swagger-UI only works in dev and test mode.

```
./mvnw compile quarkus:dev
```

quarkus.io - OpenAPI SwaggerUI Guide

Bookstore gennemgang

Author JAX-RX + service

Book JAX-RX + extends PanacheEntity

Publisher RestController/RequestMapping (Spring) + service

Test @QuarkusTest

Appendix

```
export
TOKEN="eyJraWQiOiIvcHJpdmF0ZUtleS5wZW0iLCJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJpc3MiOi
JodHRwczovL3F1YXJrdXMuaW8vdXNpbmctand0LXJiYWMiLCJqdGkiOiJhLTEyMyIsInN1YiI6Im1pY2JuLXVz
aW5nLWp3dC1yYmFjIiwidXBuIjoibWljYm5AcmVkcGlsbC1saW5wcm8uY29tIiwicHJlZmVycmVkX3VzZXJuYW
1lIjoibWljYm4iLCJhdWQiOiJ1c2luZy1qd3QtcmJhYyIsImJpcnRoZGF0ZSI6IjIwMDEtMDctMTMiLCJyb2xl
TWFwcGluZ3MiOnsiZ3JvdXAxIjoiR3JvdXAxTWFwcGVkUm9sZSIsImdyb3VwMiI6Ikdyb3VwMk1hcHBlZFJvbG
UifSwiZ3JvdXBzIjpbIkVjaG9lciIsIlRlc3RlciIsIlN1YnNjcmliZXIiLCJncm91cDIiLCJ1c2VyIl0sImlh
dCI6MTU3NjYwNjM5NiwiYXV0aF90aW1lIjoiTnVtZXJpY0RhdGV7MTU3NjYwNjM5NiAtPiAxNy1EZWMtMjAxOS
AxOToxMzoxNiBDRVR9IiwiZXhwIjoxNTc2NjA2Njk2fQ.JTgLJ7hZbkorNEg0OTyNPNAQ8_7v23ElXX2oZNBWF
-19I6k0BcnIPnydHs4Dylj3JsFtgeWSvhniv4dcEwBGphqLWKor_hG305Xv5YDiWvj0o
-9xhhEDALRvO5ycjXwt69E8YN-YA
-GdYgLyLhStSUZFjN3vMWQTO2U9UaO8jLcwZ239Z7rtw5XFy3vC9TxvDlGaPOiyBJSwagpi2oqIcdvlTHW6pB
-J3lM0B5gheHlM2btHYnqFKsviCI5Jvo9TziM8Zm1rPi4xq0yZn8Es5zVLMvrOpji5pqlCC3TkSKSomnRCC3NF
KLmxGCYYM_m9KRJWGbJCsmACY30ptM6o2A"

curl -v http://localhost:8080/todos \
    -H 'Accept: application/json' \
    -H 'Authorization: Bearer '$TOKEN'' \
    -H 'Content-Type: application/json' \
    -d '{"subject":"Hello from Quarkus","body":"Content","priority": 1,"importens":
10,"owner" : "Duke"}'

curl -v http://localhost:8080/todos \
    -H 'Accept: application/json' \
    -H 'Authorization: Bearer '$TOKEN'' \
    -H 'Content-Type: application/json' \
    -d '{"subject":"Hello from Quarkus 2","body":"Content 2","priority":
1,"importens": 10,"owner" : "Duke"}'
```

```
docker network inspect demo-net
```

```
docker network create demo-net
```

```
docker run --ulimit memlock=-1:-1 \
    -it --rm=true --memory-swappiness=0 \
    --name PostgreSQLDemo \
    --network demo-net \
    -e POSTGRES_USER=hibernate \
    -e POSTGRES_PASSWORD=hibernate \
    -e POSTGRES_DB=hibernate_db \
    -p 5432:5432 postgres:10.5
```

```
docker build -f src/main/docker/Dockerfile.native -t quarkus-fagligfredag-demo .
```

```
docker run -i --rm --name quarkus-fagligfredag --network demo-net -p 8080:8080
quarkus-fagligfredag-demo
```

Start postgres på minikube.

```
kubectl run postgresqldemo \
    --image=postgres:10.5 \
    --port=5432 \
    --env=POSTGRES_USER=hibernate \
    --env=POSTGRES_PASSWORD=hibernate \
    --env=POSTGRES_DB=hibernate_db \
    --image-pull-policy=IfNotPresent

kubectl expose deployment postgresqldemo --type=NodePort
```

Start quarkus-fagligfredag-demo på minikube.

```
kubectl run quarkus-fagligfredag-demo \
    --port=8080 \
    --image=quarkus-fagligfredag-demo:latest \
    --image-pull-policy=IfNotPresent

kubectl expose deployment quarkus-fagligfredag-demo --type=LoadBalancer --name=quarkus
-fagligfredag-demo-service
```

```
kubectl get all
```

```
kubectl delete deployment.apps/quarkus-fagligfredag-demo
```

```
minikube dashboard --url
minikube service pgadmin --url
minikube service postgresqldemo --url
minikube service quarkus-fagligfredag-demo-service --url
```

```
eval $(minikube docker-env)
```

```
eval $(minikube docker-env -u)
```