Не Spring’ом единым:

Смотрим на Quarkus

$ whoami?

JPoint

Joker<?>

Code Fest

sysconf

podlodka

Андрей Кулешов

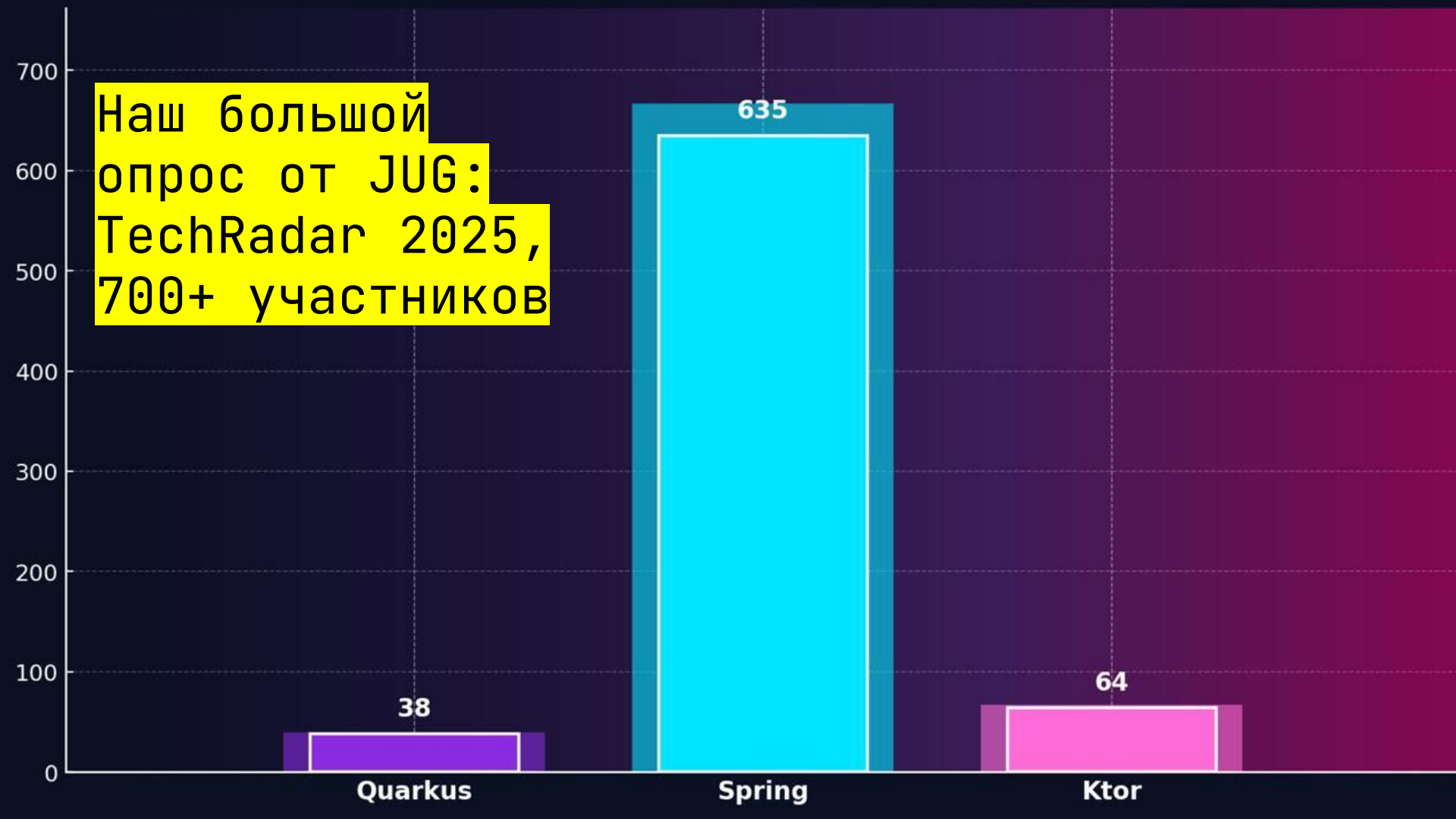
intel

Deutsche Bank

HUAWEI

pt

Я


Наш большой опрос от JUG: TechRadar 2025, 700+ участников
700
600
500
400
300
200
100
0
635
38
64
Quarkus
Spring
Ktor

«Стена плача» Java. На что хотят перейти

- Не подходит для фронта, HFT, реалтайма
- Показывает себя так себе
- Отказаться от Kotlin в пользу Java
- с Java на Kotlin
- Новую Java
- с Java на Golang
- Перейти с JPA на JDBC с Kotlin+Spring на Golang
- хочу на пенсию
- с Quarkus на Spring
- со Спринга на что угодно
- На последний Spring Boot
- с Hibernate на что угодно
- Версия Java — SberJDK
- 😱 Play Framework
- 😱 PowerMock
- Слезть «с иглы» Spring

Кирилл Толкачев
tolkv

«Стена плача» Java. На что хотят перейти

Не подходит для фронта, HFT, реалтайма

Показывает себя так себе

Отказаться от Kotlin в пользу Java

с Java на Kotlin

Новую Java

с Java на Golang

Перейти с JPA на JDBC с Kotlin+Spring на Golang

хочу на пенсию

с Quarkus на Spring

со Спринга на что угодно

На последний Spring Boot

с Hibernate на что угодно

Версия Java — SberJDK

😱 Play Framework

😱 PowerMock

Слезть «с иглы» Spring

Кирилл Толкачев

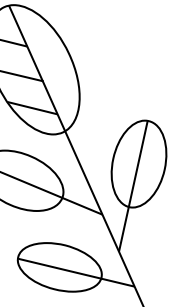
tolkv

JPoint × JUG.RU group

СБЕР

SMLab

6% опрошенных
хотели бы переехать
со Spring на Quarkus



«Стена плача» Java. На что хотят перейти

- Не подходит для фронта, HFT, реалтайма
- Показывает себя так себе
- Отказаться от Kotlin в пользу Java
- с Java на Kotlin
- Новую Java
- с Java на Golang
- Перейти с JPA на JDBC с Kotlin+Spring на Golang
- хочу на пенсию
- с Quarkus на Spring
- со Спринга на что угодно
- На последний Spring Boot
- с Hibernate на что угодно
- Версия Java — SberJDK
- 😱 Play Framework 😱 PowerMock
- Слезть «с иглы» Spring

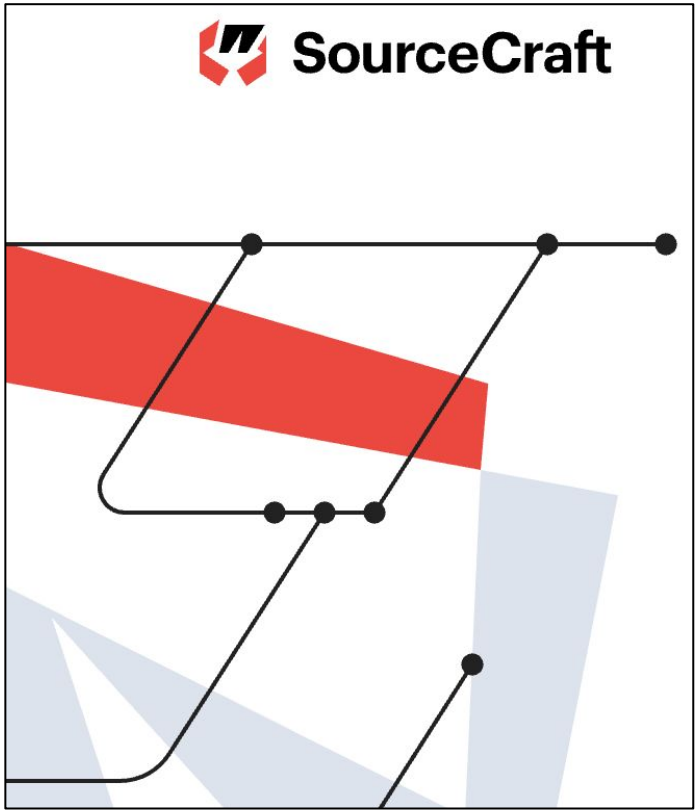
Был и один герой,
который не молчал

Кирилл Толкачев
tolkv

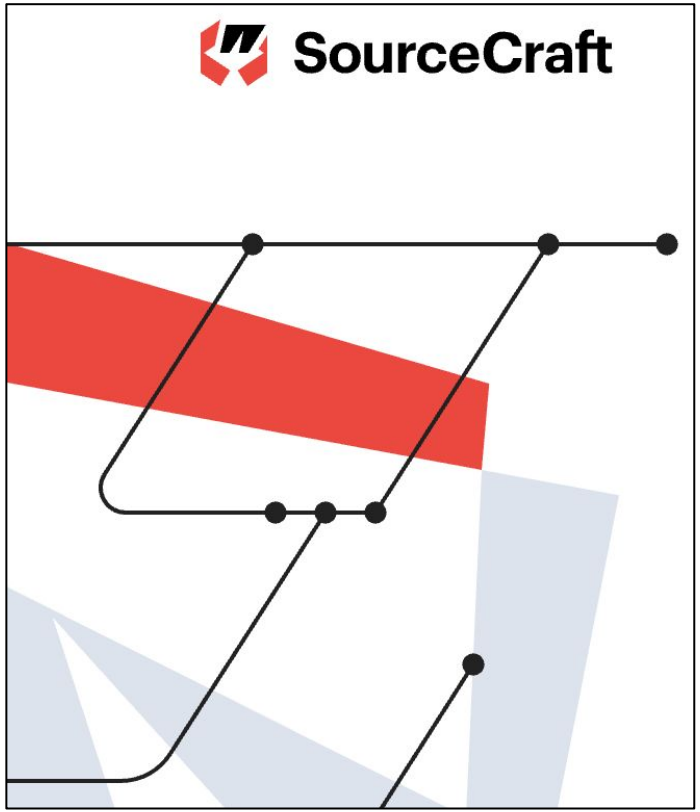
JPoint × JUG.RU group

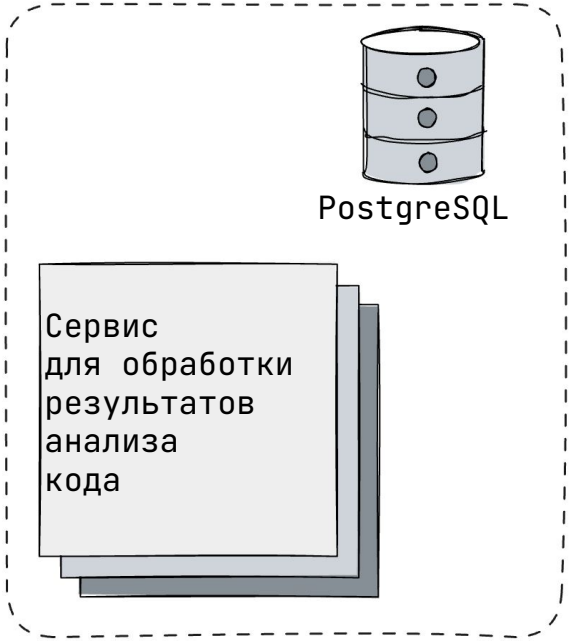
СБЕР SMLab

Моя история

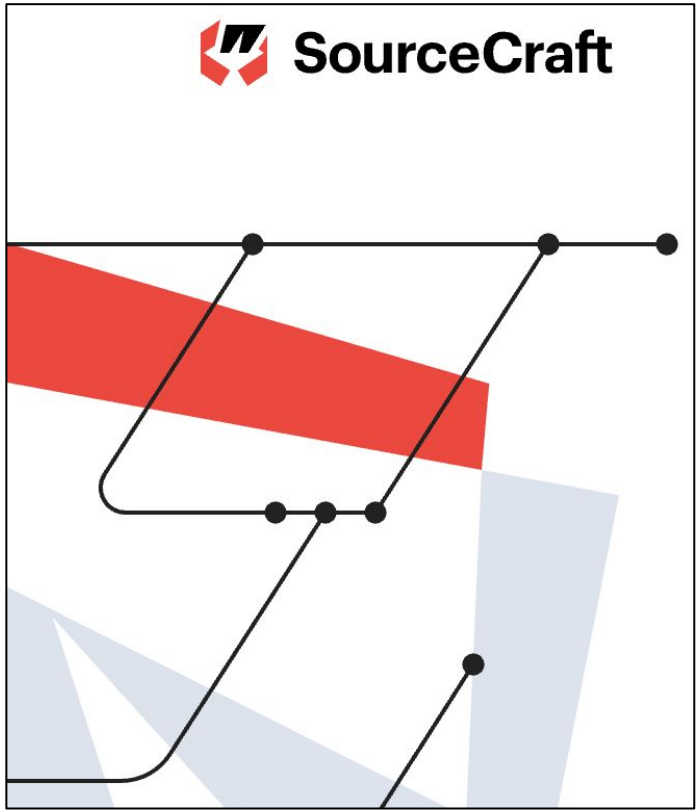


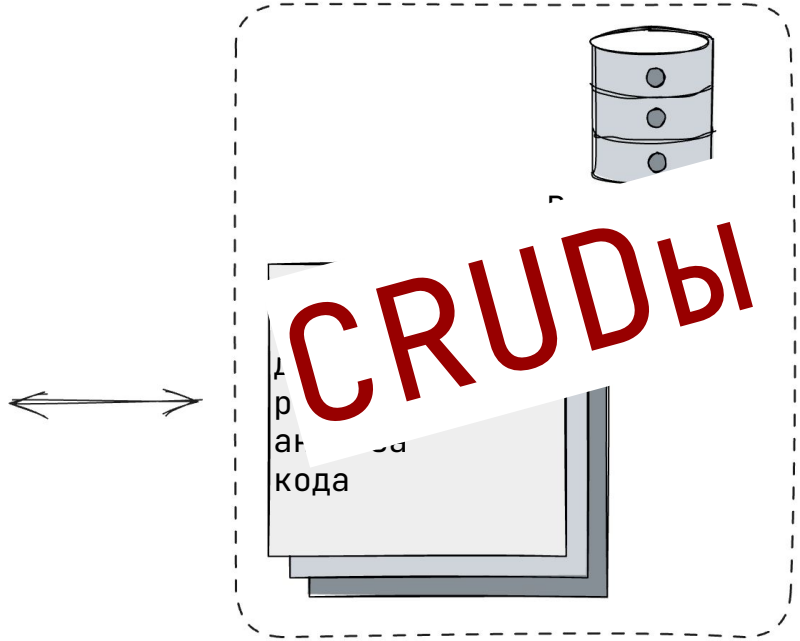
Моя история


SourceCraft


PostgreSQL
Сервис
для обработки
результатов
анализа
кода

Моя история





РАБОТАЕТ — НЕ ТРОЖЬ

Слезть «с иглы» Spring

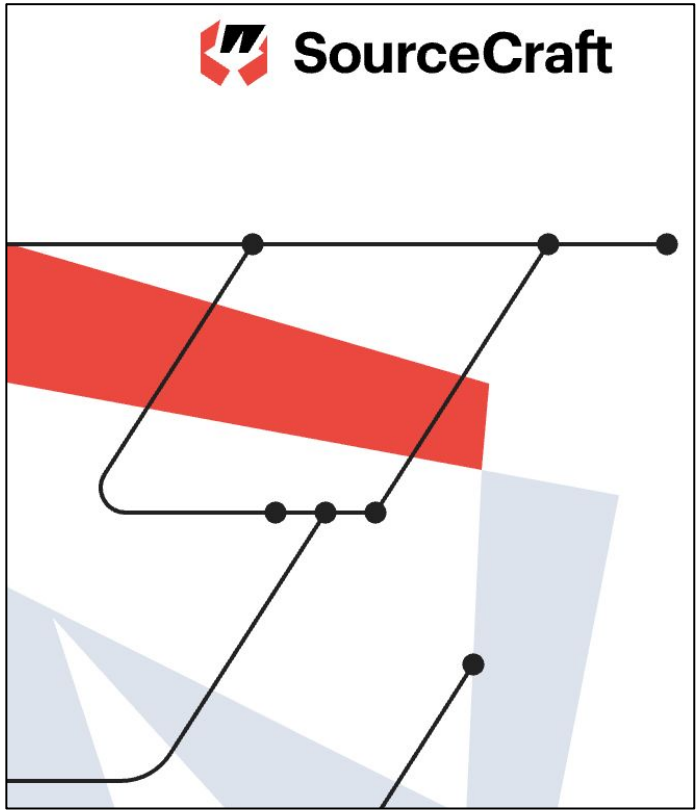
РАБОТАЕТ – ~~НЕ ТРОЖЬ~~

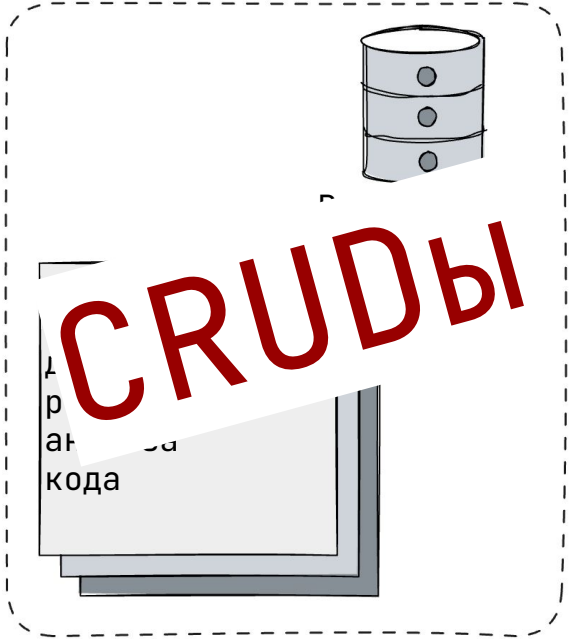
Слезть «с иглы» Spring

РАБОТАЕТ – ~~НЕ ТРОЖЬ~~

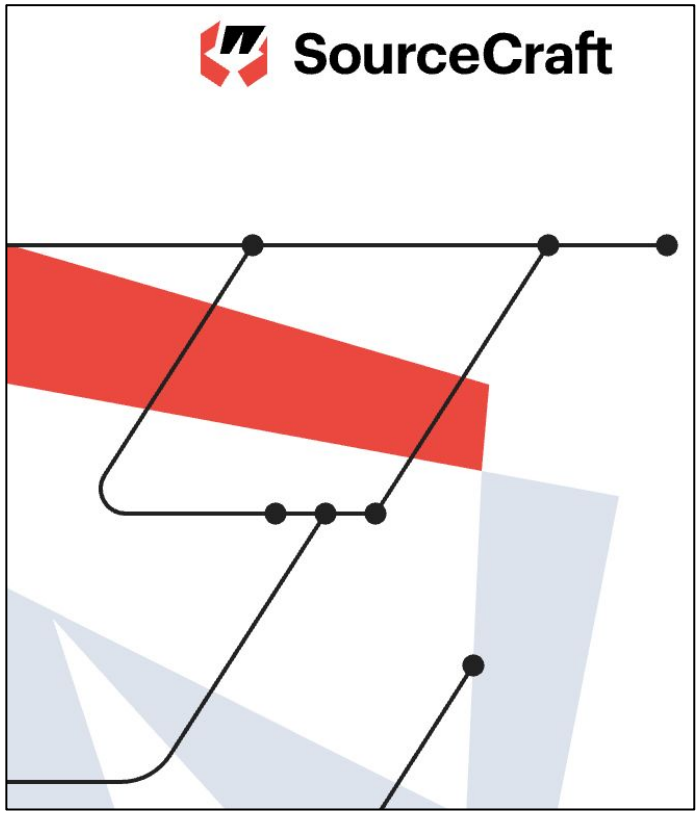
+

Моя история

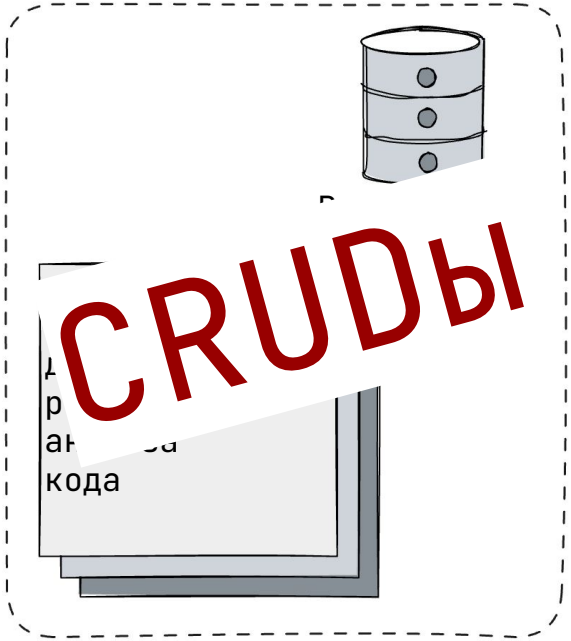




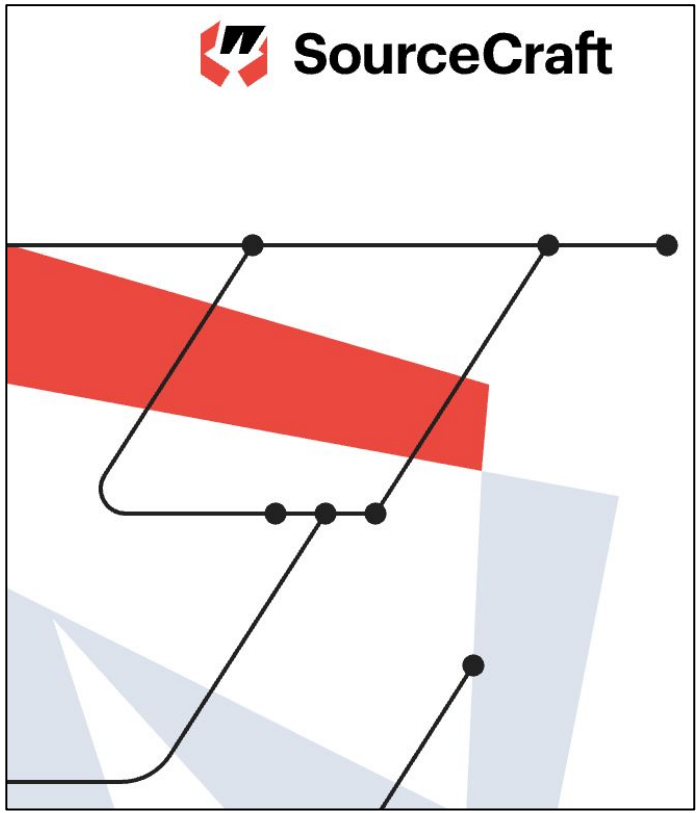
Моя история



gRPC

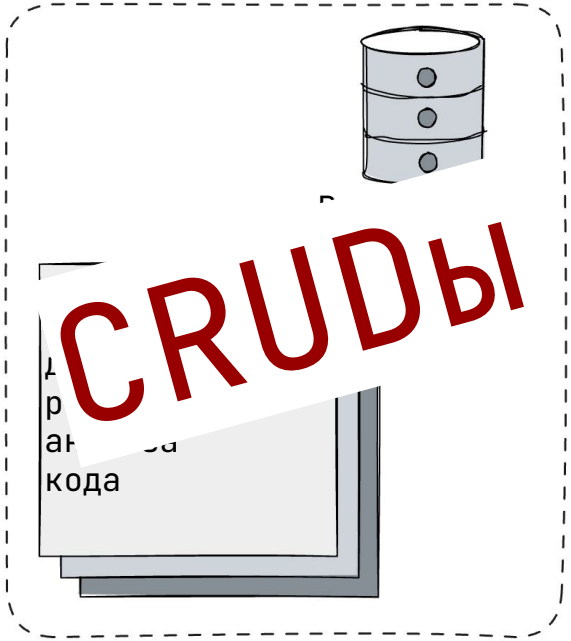


Моя история

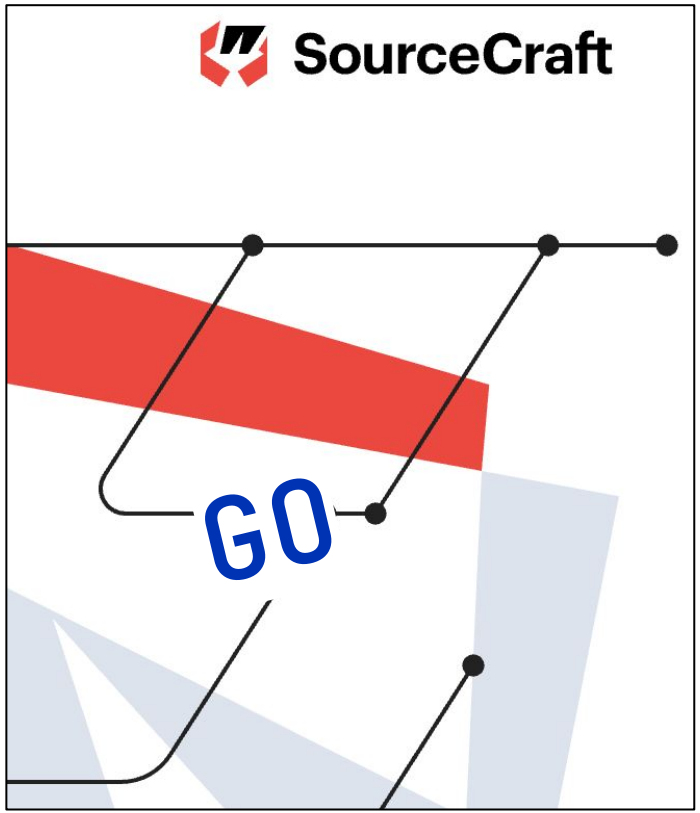


gRPC

REST

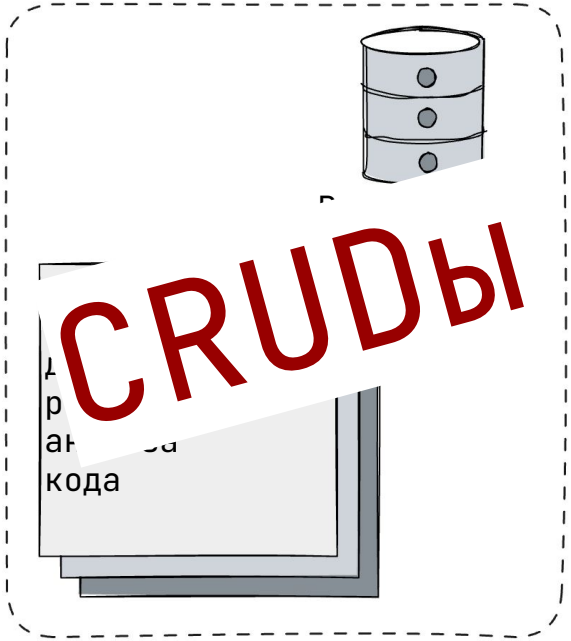


Моя история



gRPC

REST



Отличная экосистема

Мне очень нравится - отличная поддержка и активное коммьюнити

GraalVM же, натив!

Оч хороший фреймворк, а спринг - ну его

Всё из коробки же, очень enterprise-ready

Шаг в сторону - начинаются проблемы

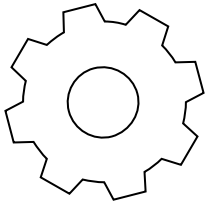
Мы все сервисы переводим на Quarkus сейчас, у нас в компании стандарт де-факто

Точно будет конфетно-букетный период, а потом…

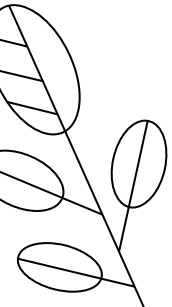
Завязывают на всё своё, редхатовское

Очень уж они заточены как-будто под быстрые промышленные сценарии

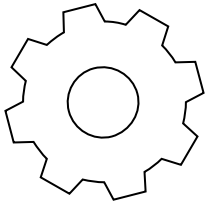
Про что поговорим?



- Мой личный опыт использования

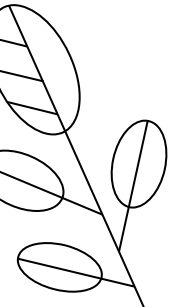


Про что поговорим?

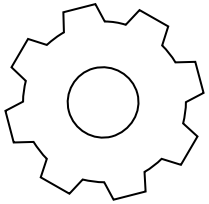


- Мой личный опыт использования

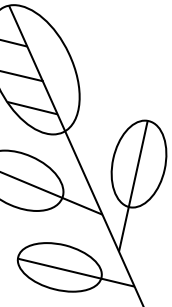
- Базовые вещи и моя **субъективная** оценка UX, не стоит ждать революционности



Про что поговорим?

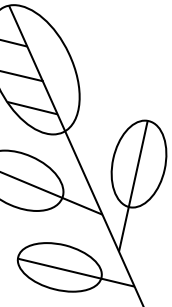


- Мой личный опыт использования
- Базовые вещи и моя **субъективная** оценка UX, не стоит ждать революционности
- **Нет** цели никого убеждать срочно переходить на Quarkus



Дисклеймер: последний раз объясняю, что не буду потрошить никого

Знакомство



Для тех, кто хочет прямых сравнений

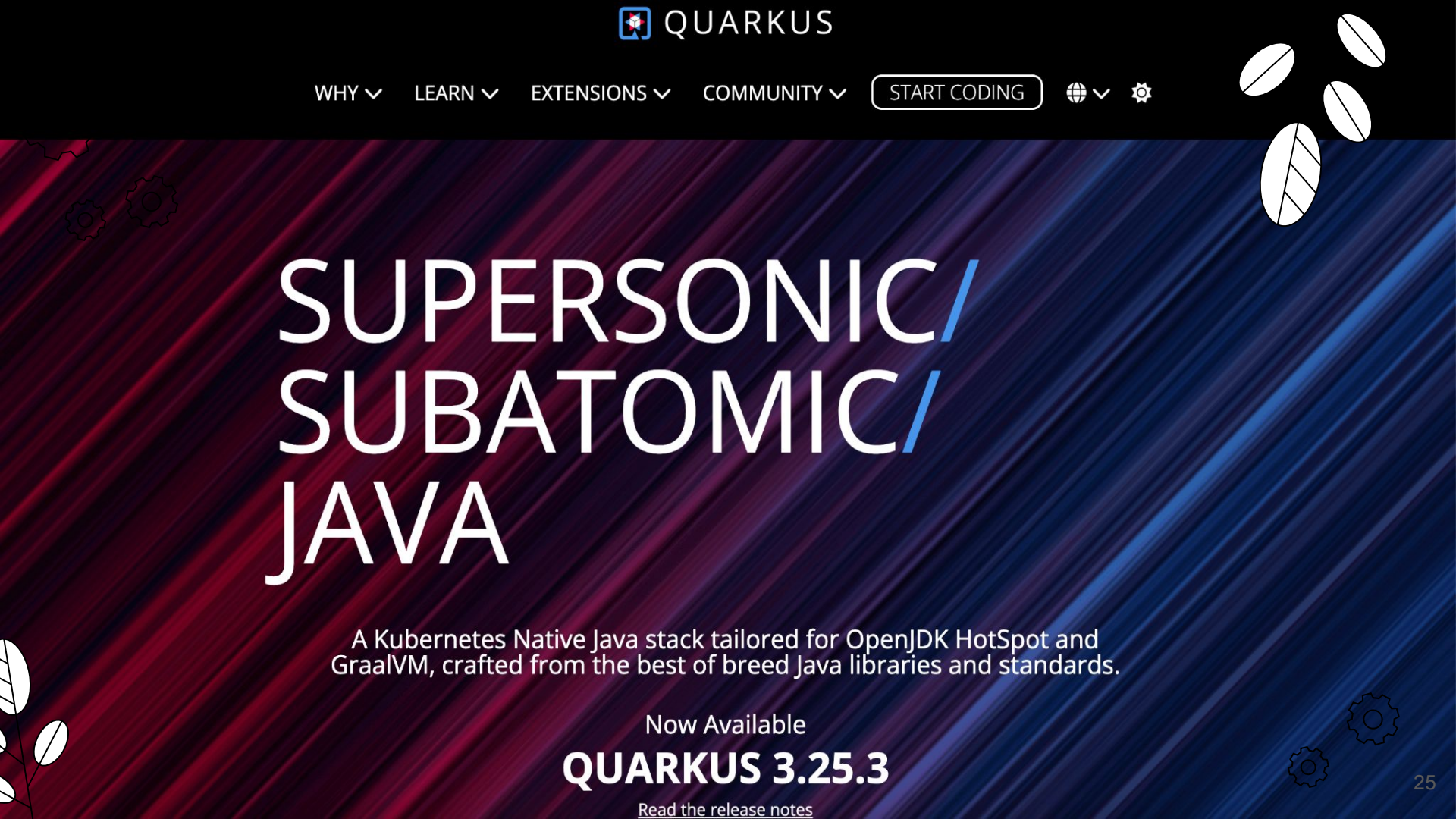
DEVOXX United Kingdom

Round up

Getting started	Database access
Rest endpoints	Websocket client
Websocket	Virtual Threads
Scheduling & startup	Dev experience
Rest client & startup-hook	Native image

35:12 / 39:23 • Bonus Round >

Spring Boot vs Quarkus the ultimate battle! by BERWOUT DE VRIES ROBLES & JAGO DE VR


QUARKUS
WHY
LEARN
EXTENSIONS
COMMUNITY
START CODING
SUPERSONIC/
SUBATOMIC/
JAVA
A Kubernetes Native Java stack tailored for OpenJDK HotSpot and GraalVM, crafted from the best of breed Java libraries and standards.
Now Available
QUARKUS 3.25.3
Read the release notes

QUARKUS

WHY LEARN EXTENSIONS COMMUNITY START CODING

SUPERSONIC/ SUBATOMIC/ JAVA

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

Now Available

QUARKUS 3.25.3

Read the release notes

Азы и терминология

Netty (HTTP/Сетевой слой)

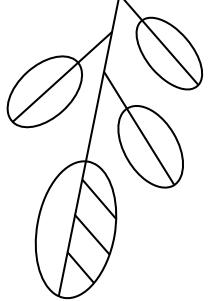
Азы и терминология

Vert.x (EventLoop/Worker Pool/итд.)

Netty (HTTP/Сетевой слой)

Азы и терминология

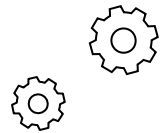
RESTEasy - реализация JAX-RS/Jakarta REST



RESTEasy

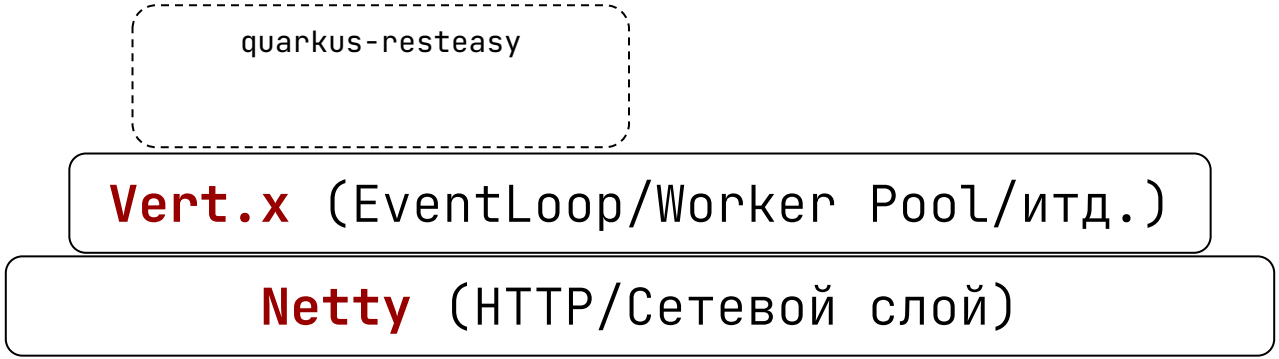
Vert.x (EventLoop/Worker Pool/итд.)

Netty (HTTP/Сетевой слой)



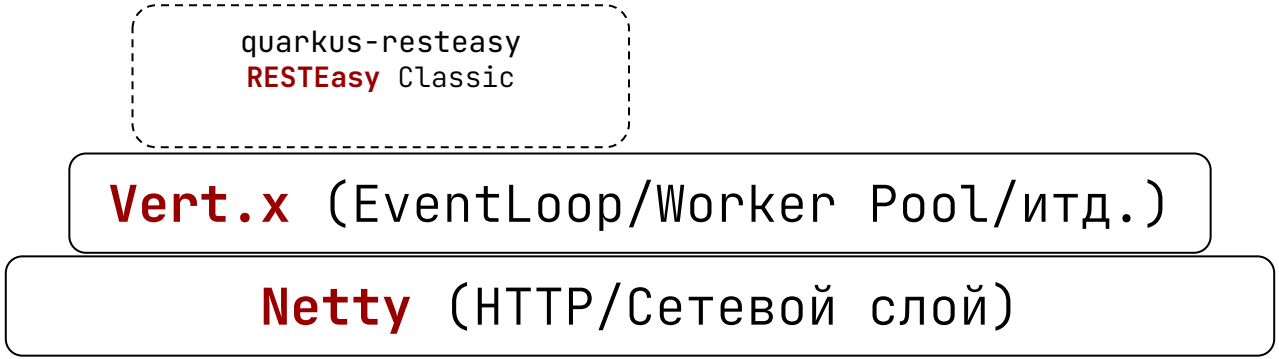
Азы и **терминология**

Default Quarkus 2.8-



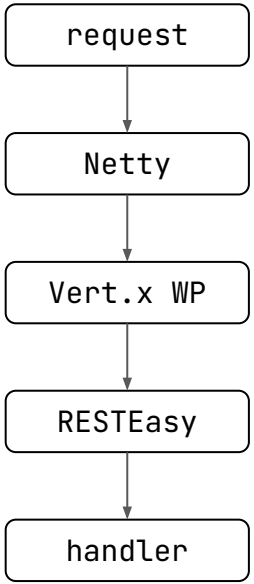
Азы и **терминология**

Default Quarkus 2.8-



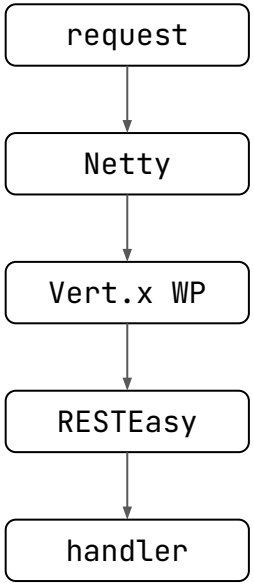
Азы и терминология

quarkus-resteasy



Азы и терминология

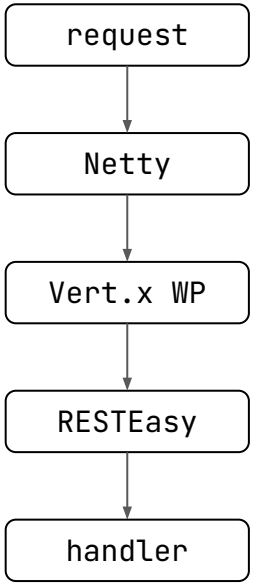
quarkus-resteasy




```
// Берется поток из WP под запрос
@Path("/users")
public class UserResource {
    @GET
    public List<User> getUsers() {



    }
}
```

Азы и терминология

quarkus-resteasy




```java
// Берется поток из WP под запрос
@Path("/users")
public class UserResource {
    @GET
    public List<User> getUsers() {
        // Императивно выполняется в worker thread
        // Поток заблокирован до завершения операции
        // Блокирующий вызов БД
        return userService.findAll();
    }
}
```

Азы и терминология

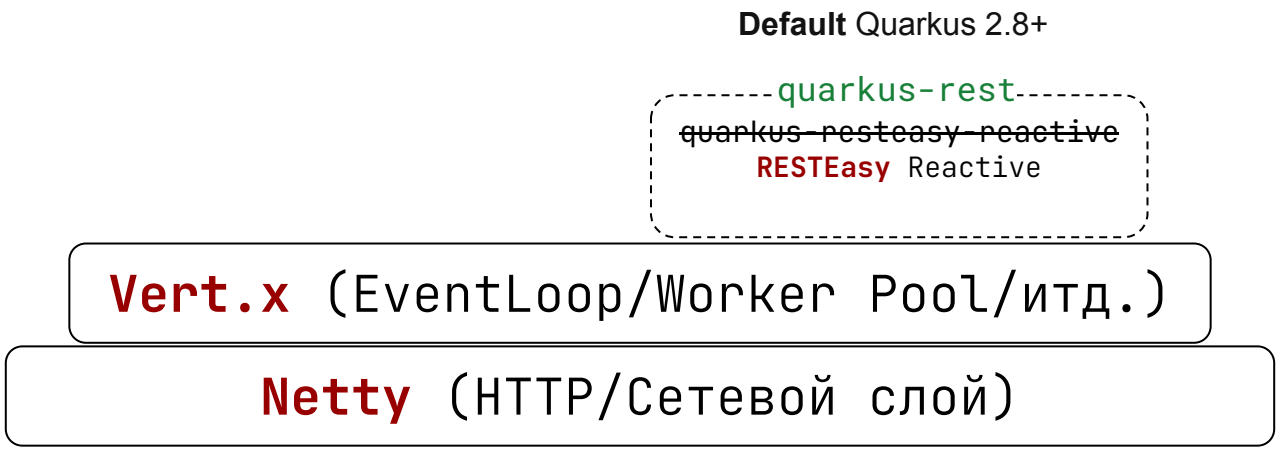
Default Quarkus 2.8+

quarkus-resteasy-reactive
RESTEasy Reactive

Vert.x (EventLoop/Worker Pool/итд.)

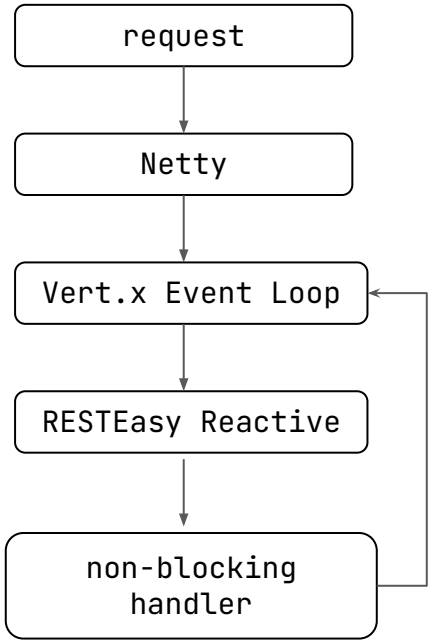
Netty (HTTP/Сетевой слой)

Азы и **терминология**



Азы и терминология

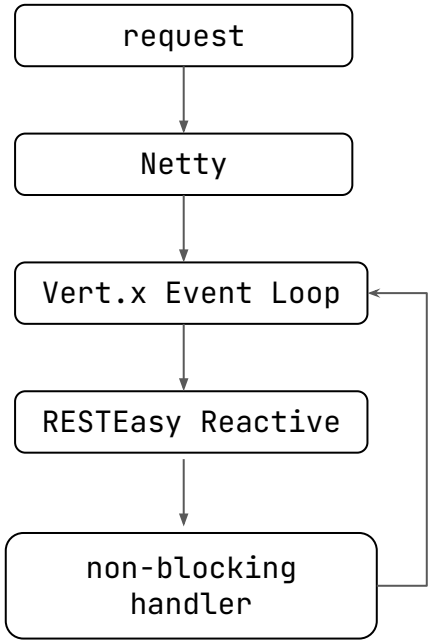
quarkus-rest(easy-reactive)




```
@Path("/users")
public class UserResource {
    @GET
    public Uni<List<String>> getUsers() {


    }
}
```

Азы и терминология

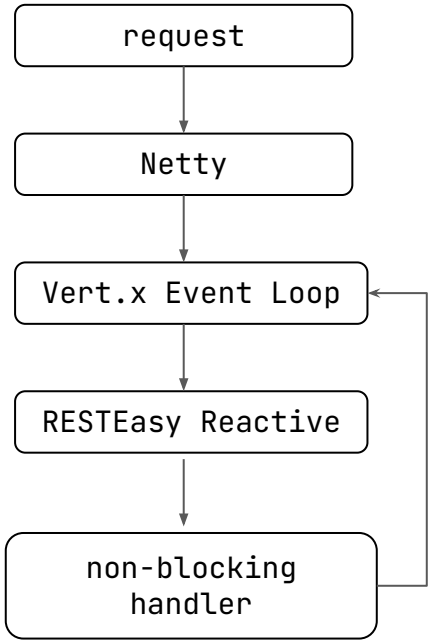
quarkus-rest(easy-reactive)




```java
@Path("/users")
public class UserResource {
    @GET
    public Uni<List<String>> getUsers() {
        // Выполняется в event loop thread
        // Поток НЕ блокируется, сразу отдает Uni<>
        return userService.findAll();
    }
}
```

Азы и терминология

quarkus-rest(easy-reactive)




```java
@Path("/users")
public class UserResource {
    @GET
    public Uni<List<String>> getUsers() {
        // Выполняется в event loop thread
        // Поток НЕ блокируется, сразу отдает Uni<>
        return userService.findAll();
    }
}
```

Реактивный фреймворк

```
package io.smallrye.mutiny;
```

Реактивный фреймворк

package io.smallrye.mutiny;

Как и project.reactor:
Event Driven + Reactive Streams

Реактивный фреймворк

package io.smallrye.mutiny;

Как и project.reactor:
Event Driven + Reactive Streams

+ По заверениям авторов проще для интеграции с Vert.x

Реактивный фреймворк

package io.smallrye.mutiny;

Как и project.reactor:
Event Driven + Reactive Streams

+ По заверениям авторов проще для интеграции с Vert.x

smallrye/mutiny

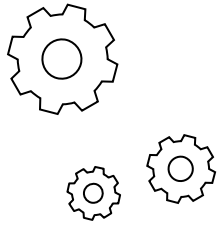
Uni

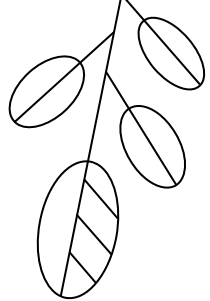
```
@GET
@Path("/user/{id}")
public Uni<User> getUser(Long id) {
   return userService.findById(id);
   {"id": 1, "name": "John"}
}
```

smallrye/mutiny

Multi

```java
// Multi - streaming response (SSE или chunked)
@GET
@Path("/users/stream")
@Produces(MediaType.SERVER_SENT_EVENTS)
public Multi<User> streamUsers() {
    return userService.streamAll();

    // data: {"id": 1, "name": "John"}
    //
    // data: {"id": 2, "name": "Jane"}
    //
    // data: {"id": 3, "name": "Bob"}
    // ...
}
```

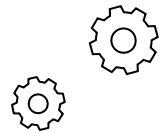




Азы и терминология

smallrye/mutiny

Quarkus’у на этапе build-time важна сигнатура метода



Азы и терминология

smallrye/mutiny

Quarkus’у на этапе build-time важна сигнатура метода

если Uni/Multi → помечается как ***non-blocking*** *route* → вешается на event-loop (**I/O thread**)

Азы и терминология

smallrye/common

@Blocking/@NonBlocking

```
@GET
 Ничего не указываем
public String getTest() {

System.out.println(Thread.currentThread().getName());
   return "Test";
}
```

Азы и терминология

smallrye/common

@Blocking/@NonBlocking

```
@GET
 Или явно @Blocking
public String getTest() {

System.out.println(Thread.currentThread().getName());
   return "Test";
}
```

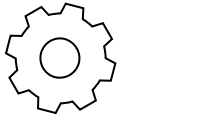
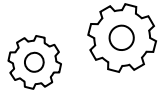
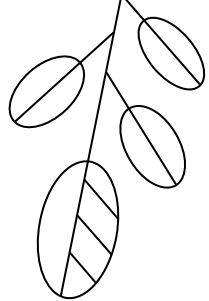
Азы и терминология

smallrye/common

@Blocking/@NonBlocking

```
@GET
 Или явно @Blocking
public String getTest() {

System.out.println(Thread.currentThread().getName());
   return "Test";
}
```

executor-thread-1

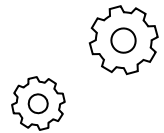




Азы и терминология

smallrye/common

@Blocking/@NonBlocking

```
@GET
@NonBlocking
public String getTest() {

System.out.println(Thread.currentThread().getName());
   return "Test";
}
```

Азы и терминология

smallrye/common

@Blocking/@NonBlocking

Или Uni/Multi возвращаемое значение без NonBlocking

```
@GET
@NonBlocking
public String getTest() {

System.out.println(Thread.currentThread().getName());
   return "Test";
}
```

Азы и терминология

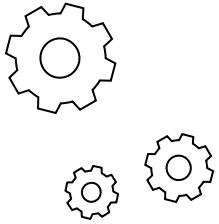
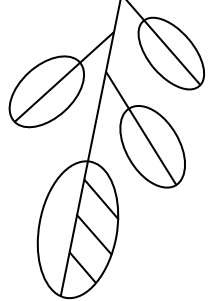
smallrye/common

@Blocking/@NonBlocking

Или Uni/Multi возвращаемое значение без NonBlocking

```
@GET
@NonBlocking
public String getTest() {

System.out.println(Thread.currentThread().getName());
   return "Test";
}
```

vert.x-eventloop-thread-1

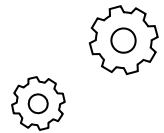



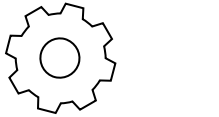
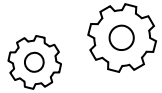
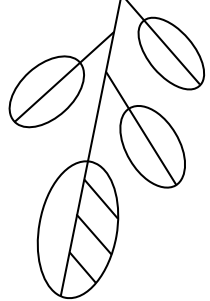
Азы и терминология

smallrye/common

@Blocking/@NonBlocking

java.lang.IllegalStateException: You have attempted to perform a blocking operation on a IO thread. This is not allowed, ***as blocking the IO thread will cause major performance issues with your application****. If you want to perform blocking EntityManager operations make sure you are doing it from a worker thread.*

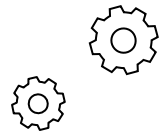




Азы и терминология

smallrye/common

@Blocking/@NonBlocking

Нет жесткого разделения моделей, как WebFlux и MVC

Азы и терминология

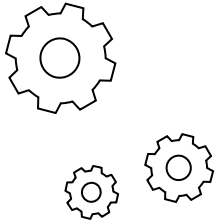
smallrye/common

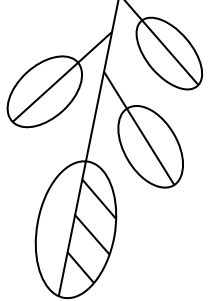
@Blocking/@NonBlocking

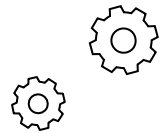
Нет жесткого разделения моделей, как WebFlux и MVC

Method signature	Dispatching strategy
`T method(…)`	Worker thread
`Uni<T> method(…)`	I/O thread
`CompletionStage<T> method(…)`	I/O thread
`Multi<T> method(…)`	I/O thread
`Publisher<T> method(…)`	I/O thread
`@Transactional CompletionStage<T> method(…)`	Worker thread

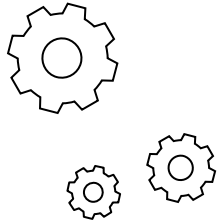
@RunOnVirtualThread

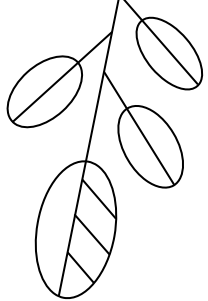




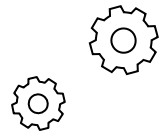


@RunOnVirtualThread





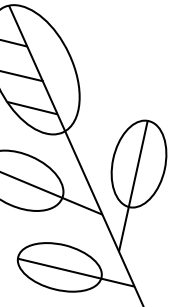
As mentioned above, not everything can run safely on virtual threads. The risk of **monopolization** can lead to **high-memory usage**. Also, there are situations where the virtual thread cannot be unmounted from the carrier thread. This is called **pinning**. Finally, some libraries use **ThreadLocal** to store and reuse objects.Using virtual threads with these libraries will lead to massive allocation, as the intentionally pooled objects will be instantiated for every (disposable and generally short-lived) virtual thread.



И даже корутины...

```
@Path("suspend")
@GET
suspend fun suspendHello(): Person {
    return Person("akuleshov7")
}
```

Быстро?



JSON serialization | Single query | Multiple queries | Cached queries | Fortunes | Data updates | Plaintext

Multiple queries

20-queries (bar) | Data table | Latency | Framework overhead

Responses per second at 20 queries per request, (542 tests)

Rnk	Framework	Performance (higher is better)		Errors	Cls	Lng	Plt	FE	Aos	DB	Dos	Orm	IA
1	just-js	89,470	100.0%	0	Plt	JS	jus	Non	Lin	Pg	Lin	Raw	Rea
2	vertx-web-kotlinx-postgresql	89,289	99.8%	0	Mcr	Kot	ver	Non	Lin	Pg	Lin	Raw	Rea
3	salvo [postgres]	89,194	99.7%	0	Mcr	rs	rs	hyp	Lin	Pg	Lin	Raw	Rea
4	quarkus, vert.x	88,902	99.4%	0	Mcr	Jav	ver	ver	Lin	Pg	Lin	Raw	Rea
5	vertx-postgres	88,708	99.1%	0	Plt	Jav	ver	Non	Lin	Pg	Lin	Raw	Rea
6	ntex [sailfish]	88,589	99.0%	0	Mcr	rs	Non	nte	Lin	Pg	Lin	Raw	Rea
7	xitca-web	88,534	99.0%	0	Mcr	rs	Non	xit	Lin	Pg	Lin	Raw	Rea
8	viz [postgresql]	88,468	98.9%	0	Ful	rs	rs	hyp	Lin	Pg	Lin	Raw	Rea
9	xitca-web [orm]	88,296	98.7%	0	Ful	rs	Non	xit	Lin	Pg	Lin	Ful	Rea
10	ntex [async-std,db]	88,149	98.5%	0	Mcr	rs	Non	nte	Lin	Pg	Lin	Raw	Rea
11	may-minihttp	88,108	98.5%	0	Mcr	rs	rs	may	Lin	Pg	Lin	Raw	Rea
12	inverno	87,919	98.3%	0	Mcr	Jav	inv	Non	Lin	Pg	Lin	Mcr	Rea
13	axum [postgresql]	87,231											
14	h2o	86,788											

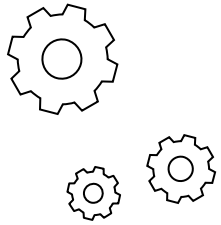
Techempower - плохой бенчмарк, но...

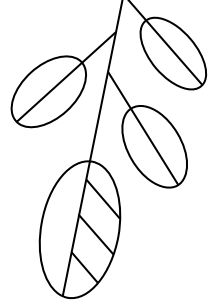
JSON serialization | Single query | Multiple queries | Cached queries | Fortunes | Data updates | Plaintext

Fortunes

Best fortunes responses per second, (523 tests)

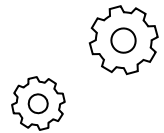
Rnk	Framework	Best performance (higher is better)		Errors	Cls	Lng	Plt	FE	Aos	DB	Dos	Orm	IA
1	may-minihttp	1,327,378	100.0%	0	Mcr	rs	rs	may	Lin	Pg	Lin	Raw	Rea
2	h2o	1,226,814	92.4%	0	Plt	C	Non	h2o	Lin	Pg	Lin	Raw	Rea
3	ntex [sailfish]	1,210,348	91.2%	0	Mcr	rs	Non	nte	Lin	Pg	Lin	Raw	Rea
4	ntex [async-std,db]	1,197,351	90.2%	0	Mcr	rs	Non	nte	Lin	Pg	Lin	Raw	Rea
5	xitca-web	1,146,712	86.4%	0	Mcr	rs	Non	xit	Lin	Pg	Lin	Raw	Rea
6	xitca-web [orm]	1,115,124	84.0%	0	Ful	rs	Non	xit	Lin	Pg	Lin	Ful	Rea
7	axum [postgresql]	1,114,265	83.9%	0	Ful	rs	rs	hyp	Lin	Pg	Lin	Raw	Rea
8	lithium-postgres	1,073,846	80.9%	0	Mcr	C++	Non	Non	Lin	Pg	Lin	Ful	Rea
9	lithium-postgres-beta	1,068,560	80.5%	0	Mcr	C++	Non	Non	Lin	Pg	Lin	Ful	Rea
10	hyper-db	1,066,644	80.4%	0	Mcr	rs	rs	hyp	Lin	Pg	Lin	Raw	Rea
11	viz [postgresql]	1,060,105	79.9%	0	Ful	rs	rs	hyp	Lin	Pg	Lin	Raw	Rea
12	drogon-core	1,042,653	78.5%	0	Ful	C++	Non	Non	Lin	Pg	Lin	Raw	Rea
13	vertx-postgres	1,040,599	78.4%	0	Plt	Jav	ver	Non	Lin	Pg	Lin	Raw	Rea
14	quarkus, vert.x	1,028,408	77.5%	0	Mcr	Jav	ver	ver	Lin	Pg	Lin	Raw	Rea





Про скорость

- Quarkus жертвует **“универсальностью”** и **“совместимостью”**



Про скорость

- Quarkus жертвует **“универсальностью”** и **“совместимостью”**

- **Берите, что дают:** ядро фреймворка ограничено тем, что удобно разработчикам, не распухает

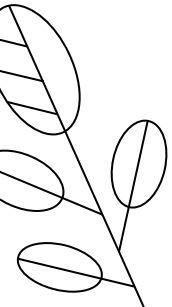
Про скорость

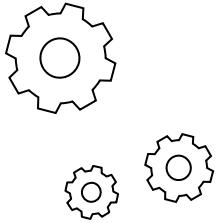
- Quarkus жертвует **“универсальностью”** и **“совместимостью”**
- **Берите, что дают**: ядро фреймворка ограничено тем, что удобно разработчикам, не распухает
- Попытка переносить бОльшую часть логики на build time (например, **DI**)

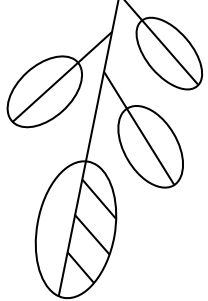
CDI

jakarta.enterprise.cdi

Arc



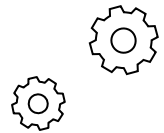


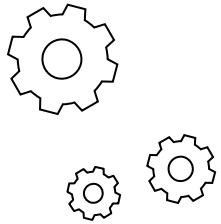


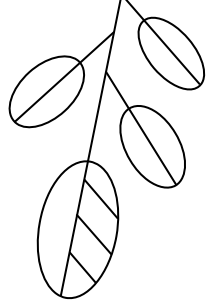
Азы и терминология

Arc - приблизительная имплементация спецификации Jakarta Contexts and Dependency Injection 4.1

```
quarkus.arc.strict-compatibility=true
```

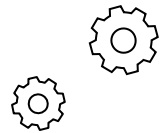


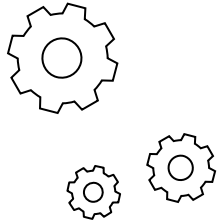


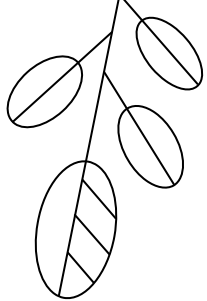


Arc - приблизительная имплементация спецификации [Jakarta Contexts and Dependency Injection 4.1]

Работает за счет **ASM**, библиотеки для манипуляций с байткодом



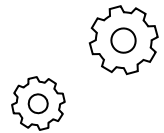




Arc - приблизительная имплементация спецификации [Jakarta Contexts and Dependency Injection 4.1]

Работает за счет **ASM**, библиотеки для манипуляций с байткодом

Андрей Кулешов: "Компиляторные плагины в Котлин: почувствуй себя системным программистом"



Contexts and Dependency Injection

```
@ApplicationScoped
public class UserService {
    public User findById(Long id) {
        return new User(id, "John");
    }
}
```

Singleton, с дополнительными сгенерированными обертками и прокси

Contexts and Dependency Injection

```java
@ApplicationScoped
public class OrderService {

    // ===== Field или setter
    @Inject
    UserService userService;



}
```

Contexts and Dependency Injection

```
@ApplicationScoped
public class OrderService {

    // ===== Field или setter
    @Inject
    UserService userService;

    // ===== Конструктор
    private final PaymentService paymentService;

    @Inject
    public OrderService(PaymentService paymentService) {
        this.paymentService = paymentService;
    }



}
```

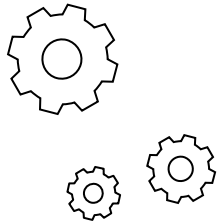
Contexts and Dependency Injection

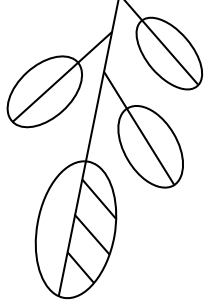
```java
@ApplicationScoped
public class OrderService {

    // ===== Field или setter
    @Inject
    UserService userService;

    // ===== Конструктор
    private final PaymentService paymentService;

    @Inject
    public OrderService(PaymentService paymentService) {
        this.paymentService = paymentService;
    }

    // ===== Метод
    @Inject
    void configure(ConfigService config, LogService log) {
        // Вызывается после создания объекта
    }
}
```

Contexts and Dependency Injection

```
@RequestScoped
public class RequestContext {
   private String requestId =
           UUID.randomUUID().toString();

   public String getRequestId() {
       return requestId;
   }
}
```

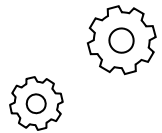
Новый экземпляр, например для каждого HTTP запроса



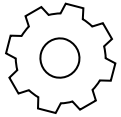


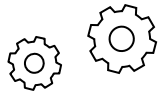
`@Dependent` - создается новый экземпляр каждый раз при инъекции

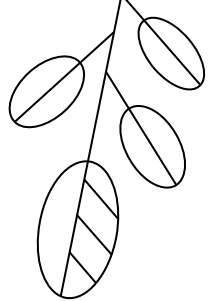
`@Singleton` - похож на `@ApplicationScoped`, но не создается proxy



ARC / Jakarta **Interceptors**







@Transactional

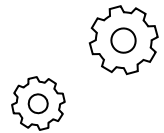
@CircuitBreaker

@RateLimit

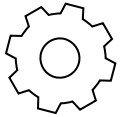
@Timeout

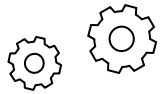
@Retry

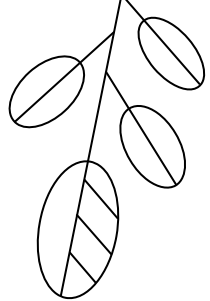
@Fallback



ARC / Jakarta Interceptors







@Transactional

Arc во время **build-time**:

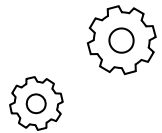
@CircuitBreaker

@RateLimit

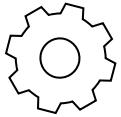
@Timeout

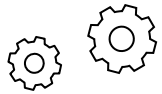
@Retry

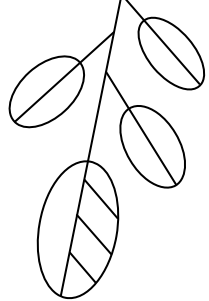
@Fallback



ARC / Jakarta Interceptors







@Transactional

@CircuitBreaker

@RateLimit

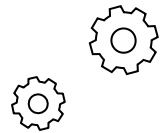
@Timeout

@Retry

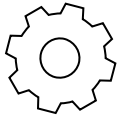
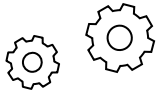
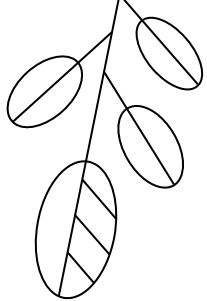
@Fallback

Arc во время build-time:

- Находит всё, что имеет аннотации-интерсепторы итд



ARC / Jakarta Interceptors

@Transactional

@CircuitBreaker

@RateLimit

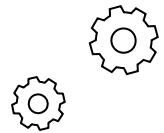
@Timeout

@Retry

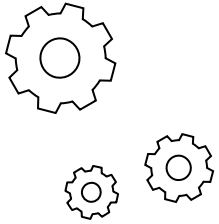
@Fallback

Arc во время build-time:

- Находит всё, что имеет аннотации-интерсепторы итд
- Генерирует байткод, который не вызывает метод напрямую, а вставляет вызов цепочки интерсепторов

ARC / Jakarta Interceptors



@Transactional

@CircuitBreaker

@RateLimit

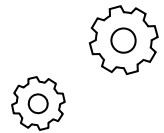
@Timeout

@Retry

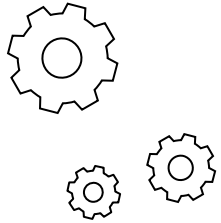
@Fallback

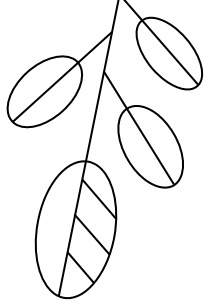
<u>Arc во время build-time:</u>

- Находит всё, что имеет аннотации-интерсепторы итд
- Генерирует байткод, который не вызывает метод напрямую, а вставляет вызов цепочки интерсепторов
- В runtime, при вызове метода, сначала выполняются интерсепторы, потом и сам метод



ARC / Jakarta Interceptors





@Transactional

@RateLimit

@CircuitBreaker

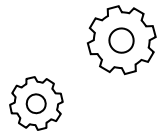
@Timeout

@InterceptorBinding

@RegisterInterceptor

@Fallback

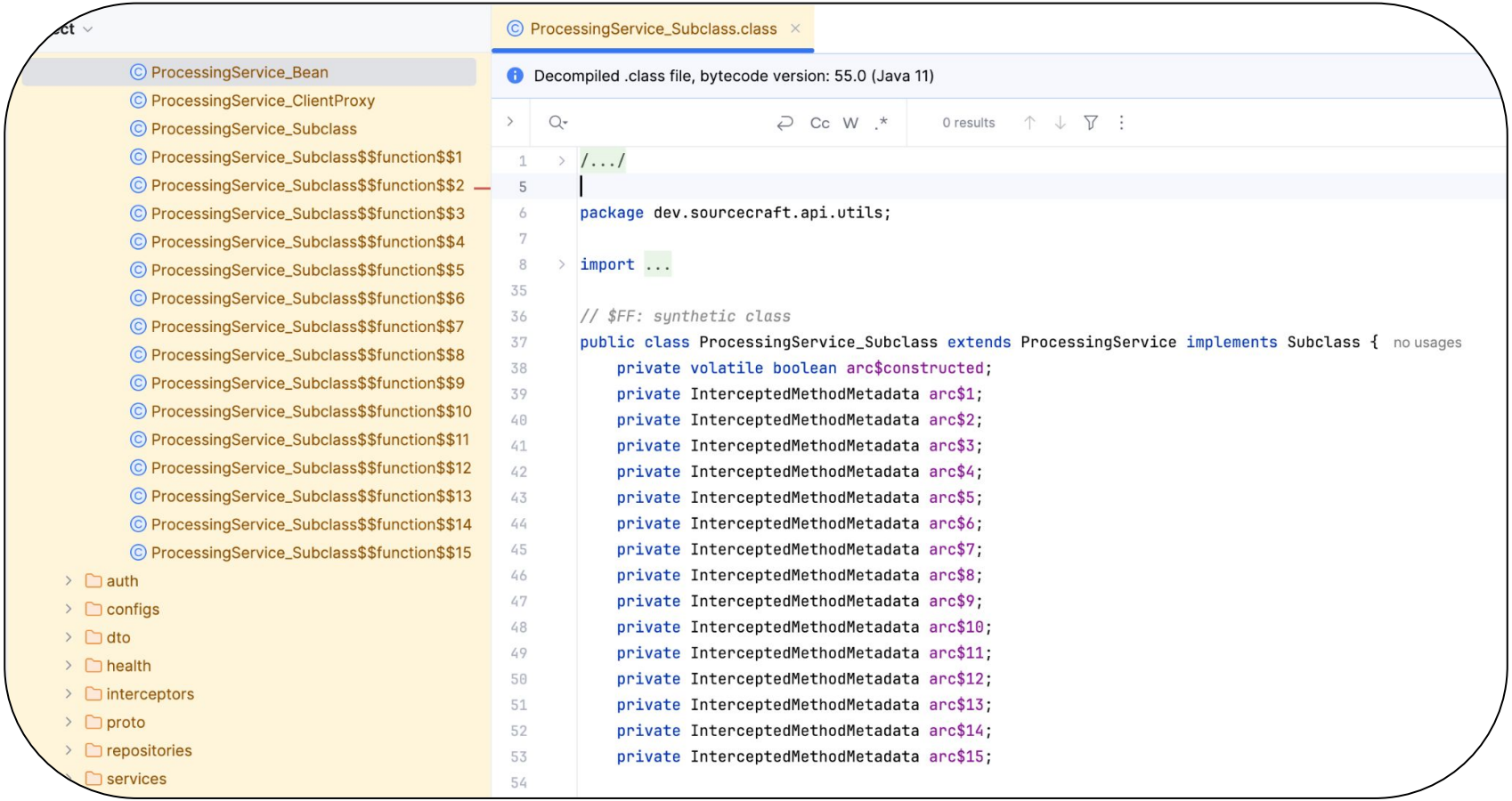
@Retry



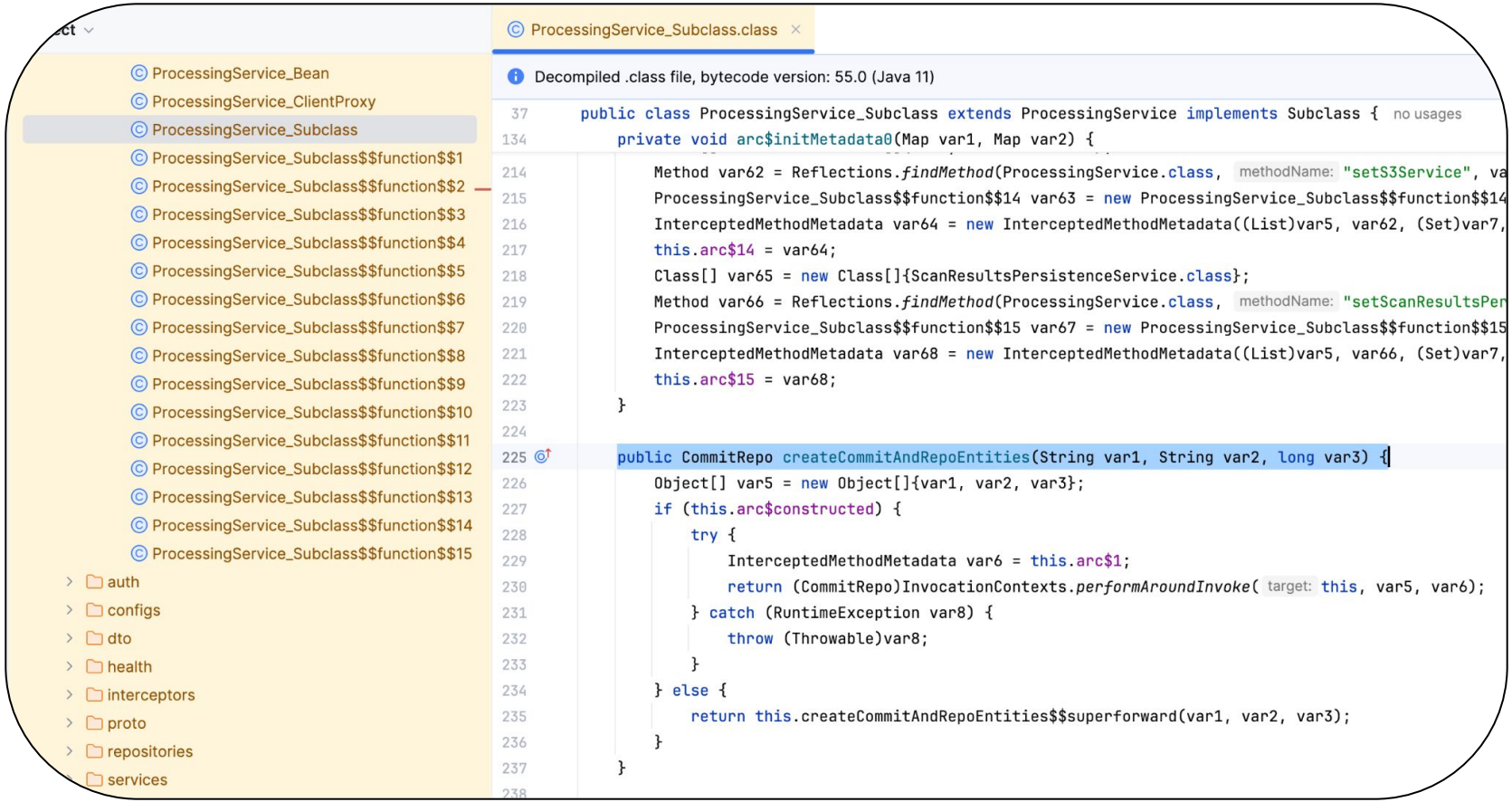
Bytecode-gen

```
@ApplicationScoped
class ProcessingService {
    fun createCommitAndRepoEntities() {
            ...
     }
}
```

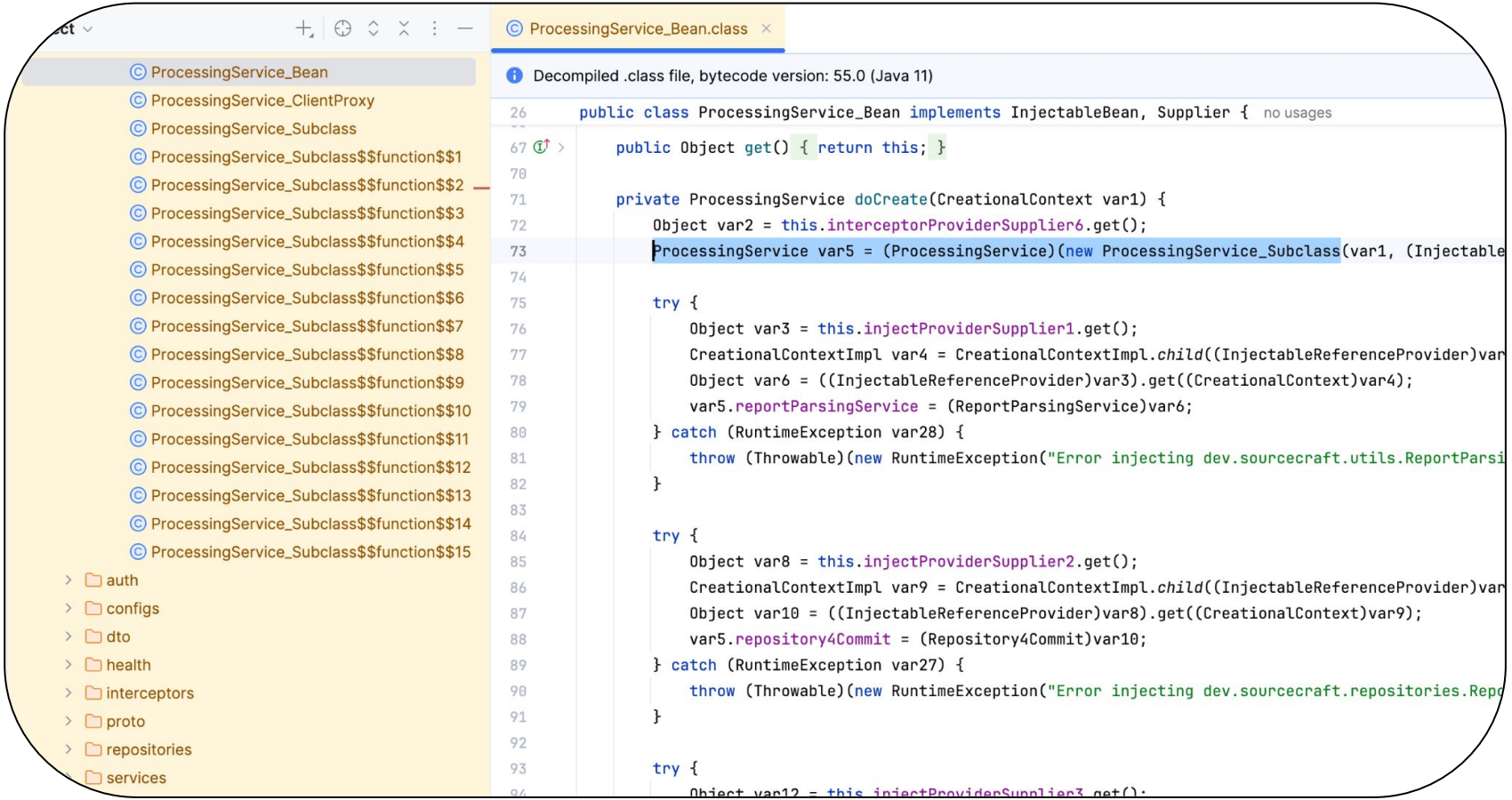
Subclass - реальный объект



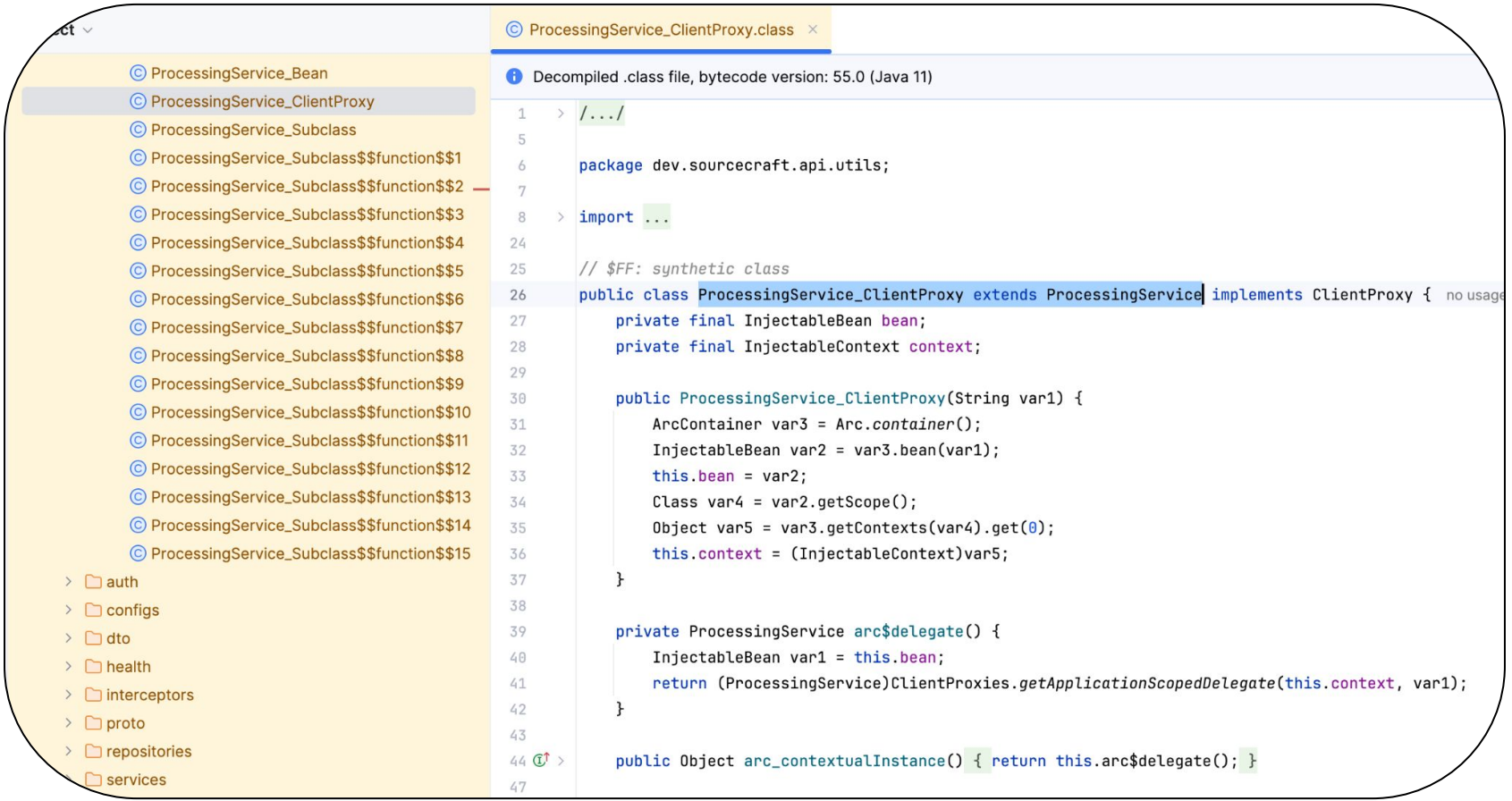
Subclass - интерсепторы в логике

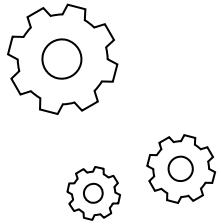
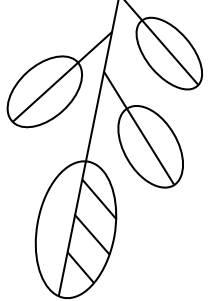


Bean-конструктор

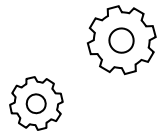


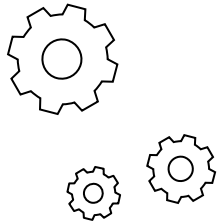
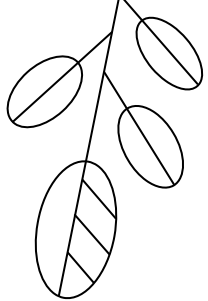
Client-proxy



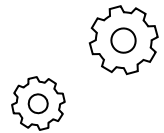



META-INF/jandex.idx

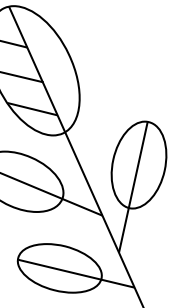



@Unremovable

Что отдаленно напоминает?



подсказка

JEP 483: AOT Class Loading & Linking

JEP 514: AOT Command-Line Ergonomics

JEP 483: AOT Class Loading & Linking

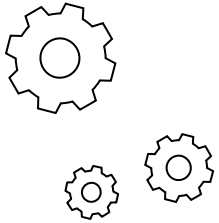
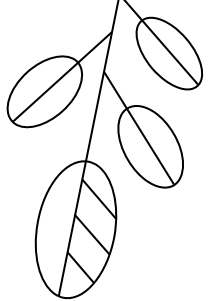
JEP 514: AOT Command-Line Ergonomics

- AOT cache
- *Не требуется обнаруживать, линковать, отдельно загружать классы на старте*

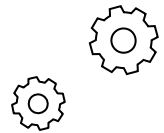
JEP 483: AOT Class Loading & Linking

JEP 514: AOT Command-Line Ergonomics

- AOT cache
- *Не требуется обнаруживать, линковать, отдельно загружать классы на старте*
- *Spring PetClinic* 📈 ***42%*** *(21k classes)*
- *Поможет ли Quarkus'у?!*

Вкатываемся в UX!

Что мне сразу бросилось в глаза?

```
% quarkus create && cd code-with-quarkus
Creating an app (default project type, see --help).
Looking for the newly published extensions in registry.quarkus.io

-----------

applying codestarts...
📚 java
🔨 maven
📦 quarkus
📝 config-properties
🔧 tooling-dockerfiles
🔧 tooling-maven-wrapper
🚀 rest-codestart

-----------
[SUCCESS] ✅  quarkus project has been successfully generated in:
⟶ /Users/akuleshov7/projects/code-with-quarkus
```

./mvnw quarkus:dev

```
__  ____  __  _____   ___  __ ____  ______
 --/ __ \/ / / / _ | / _ \/ //_/ / / / __/
 -/ /_/ / /_/ / __ |/ , _/ ,< / /_/ /\ \
--\___\_\____/_/ |_/_/|_/_/|_|\____/___/
2025-08-19 23:59:46,859 INFO  [io.quarkus] (Quarkus Main Thread) code-with-quarkus 1.0.0-SNAPSHOT on JVM (powered by Quarkus 3.25.3)
2025-08-19 23:59:46,860 INFO  [io.quarkus] (Quarkus Main Thread) Profile dev activated. Live Coding activated.
2025-08-19 23:59:46,860 INFO  [io.quarkus] (Quarkus Main Thread) Installed features: [cdi, compose, kubernetes, rest, smallrye-conte
xt-propagation, smallrye-health, vertx]

--
Tests paused
Press [e] to edit command line args (currently ''), [r] to resume testing, [o] Toggle test output, [:] for the terminal, [h] for mor
e options>
```

```
quarkus.test.continuous-testing=enabled
```

./mvnw quarkus:dev

```
--  ____  __  _____   ___  __ ____  ______
 --/ __ \/ / / / _ | / _ \/ //_/ / / / __/
 -/ /_/ / /_/ / __ |/ , _/ ,< / /_/ /\ \
--\___\_\____/_/ |_/_/|_/_/|_|\____/___/
2025-08-19 23:59:46,859 INFO  [io.quarkus] (Quarkus Main Thread) code-with-quarkus 1.0.0-SNAPSHOT on JVM (powered by Quarkus 3.25.3)
2025-08-19 23:59:46,860 INFO  [io.quarkus] (Quarkus Main Thread) Profile dev activated. Live Coding activated.
2025-08-19 23:59:46,860 INFO  [io.quarkus] (Quarkus Main Thread) Installed features: [cdi, compose, kubernetes, rest, smallrye-conte
xt-propagation, smallrye-health, vertx]

--
Tests paused
Press [e] to edit command line args (currently ''), [r] to resume testing, [o] Toggle test output, [:] for the terminal, [h] for mor
e options>
```

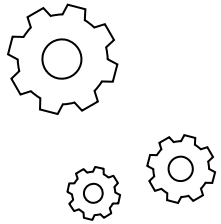
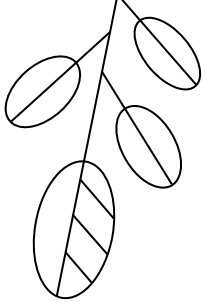
```
quarkus.test.continuous-testing=enabled
```

./mvnw quarkus:dev

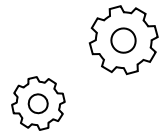
```
--  ____  __  _____   ___  __ ____  ______
 --/ __ \/ / / / _ | / _ \/ //_/ / / / __/
 -/ /_/ / /_/ / __ |/ , _/ ,< / /_/ /\ \
--\___\_\____/_/ |_/_/|_/_/|_|\____/___/
2025-08-19 23:59:46,859 INFO  [io.quarkus] (Quarkus Main Thread) code-with-quarkus 1.0.0-SNAPSHOT on JVM (powered by Quarkus 3.25.3)
2025-08-19 23:59:46,860 INFO  [io.quarkus] (Quarkus Main Thread) Profile dev activated. Live Coding activated.
2025-08-19 23:59:46,860 INFO  [io.quarkus] (Quarkus Main Thread) Installed features: [cdi, compose, kubernetes, rest, smallrye-conte
xt-propagation, smallrye-health, vertx]

--
Tests paused
Press [e] to edit command line args (currently ''), [r] to resume testing, [o] Toggle test output, [:] for the terminal, [h] for mor
e options>
```

quarkus.test.continuous-testing=enabled

\+ Hot reload, но этим пользователей Spring не удивить

Сразу куча девсервисов из коробки

Dev UI

- Extensions
- Configuration
- Workspace
- Endpoints
- Continuous Testing
- Dev Services
- Build Metrics
- Readme
- Dependencies

Configuration

Filter

Phase	Name	Value
	quarkus.analytics.disabled	
	quarkus.analytics.timeout	3000
	quarkus.analytics.uri.base	
	quarkus.application.name	code-with-quarkus
	quarkus.application.ui-header	{applicationName} (powered by Quark
	quarkus.application.version	1.0.0-SNAPSHOT
	quarkus.arc.auto-inject-fields	
	quarkus.arc.auto-producer-methods	
	quarkus.arc.context-propagation.enabled	
	quarkus.arc.detect-unused-false-positives	
	quarkus.arc.detect-wrong-annotations	
	quarkus.arc.dev-mode.generate-dependency-graphs	
	quarkus.arc.dev-mode.monitoring-enabled	
	quarkus.arc.exclude-dependency.*.artifact-id	
	quarkus.arc.exclude-dependency.*.classifier	
	quarkus.arc.exclude-dependency.*.group-id	

Dev UI

- Extensions
- Configuration
- Workspace
- Endpoints
- Continuous Testing
- Dev Services
- Build Metrics
- Readme
- Dependencies

Endpoints

Resource Endpoints

URL	Description
/hello	GET (produces:text/plain;charset=U

Additional endpoints

URL	Description
http://localhost:9003/q/arc	CDI Overview
http://localhost:9003/q/arc/beans	Active CDI Beans
http://localhost:9003/q/arc/observers	Active CDI Observers
http://localhost:9003/q/arc/removed-beans	Removed CDI Beans
http://localhost:9003/q/dev-ui	Dev UI
http://localhost:9003/q/health	Health Check
http://localhost:9003/q/health-ui	Health UI
http://localhost:9003/q/health/group	Health Group Check
http://localhost:9003/q/health/group/*	Health Group Check
http://localhost:9003/q/health/live	Health Liveness Check
http://localhost:9003/q/health/ready	Health Readiness Check
http://localhost:9003/q/health/started	Health Started Check

Dev UI

- Extensions
- Configuration
- Workspace
- Endpoints
- Continuous Testing
- Dev Services
- Build Metrics
- Readme
- Dependencies

Continuous Testing

Run all | Run failed | Only run failing tests | Display tags (if available)

Tags	Test Class	Name
Q	org.acme.GreetingResourceTest	testHelloEndpoint()

Dev UI

Workspace

- Extensions
- Configuration
- Workspace
- Endpoints
- Continuous Testing
- Dev Services
- Build Metrics
- Readme
- Dependencies

GreetingResource.java

- src
  - main
    - java
      - org
        - acme
          - GreetingResource.java
    - resources
      - application.properties
    - docker
      - Dockerfile.jvm
      - Dockerfile.legacy-jar
      - Dockerfile.native
      - Dockerfile.native-micro
  - test
    - java
      - org

```java
package org.acme;

import jakarta.ws.rs.GET;
import jakarta.ws.rs.Path;
import jakarta.ws.rs.Produces;
import jakarta.ws.rs.core.MediaType;

@Path("/hello")
public class GreetingResource {

    @GET
    @Produces(MediaType.TEXT_PLAIN)
    public String hello() {
        return "Hello from Quarkus REST";
    }
}

```

src/main/java/org/acme/GreetingResource.java saved successfully


Dev UI
Extensions
Configuration
Workspace
Endpoints
Continuous Testing
Dev Services
Build Metrics
Readme
Dependencies
Dependencies
Show path to ...
asm-util
smallrye-common-cpu
quarkus-development-mode-spi
commons-logging-jboss-logging
quarkus-class-change-agent
quarkus-bootstrap-runner
smallrye-config-core
quarkus-core
jboss-logmanager
jboss-threads
asm-tree
asm-commons
smallrye-common-expression
jandex
quarkus-builder
quarkus-core-deployment
smallrye-common-io
wildfly-common
jboss-logging
quarkus-smallrye-openapi-spi
smallrye-common-ref
smallrye-config-common
asm
smallrye-common-function
microprofile-config-api
quarkus-bootstrap-core
qute-core
smallrye-common-constraint
quarkus-jackson-spi
smallrye-common-classloader
quarkus-container-image-spi
smallrye-beanbag
smallrye-common-os
smallrye-common-process
quarkus-devservices-deployment
quarkus-datasource-common
quarkus-fs-util
quarkus-devui-deployment
aesh
quarkus-bootstrap-app-model
smallrye-open-api-core
quarkus-kubernetes-client-spi
slf4j-jboss-logmanager
smallrye-common-version
readline
smallrye-beanbag-sisu
microprofile-openapi-api
snakeyaml
quarkus-devtools-message-writer
quarkus-devservices-common
org.eclipse.sisu.inject
jackson-core
quarkus-devtools-common
commons-lang3
jackson-datatype-jdk8
kubernetes-client
quarkus-bootstrap-maven-resolver
jansi
importmap
jdom2
quarkus-devtools-codestarts
jackson-dataformat-yaml
jackson-databind

Не “паришься” по поводу локального/тестового запуска

Не “паришься” по поводу локального/тестового запуска

```
quarkus.datasource.devservices.enabled=true
quarkus.datasource.devservices.image-name=postgres:15
```

Не “паришься” по поводу локального/тестового запуска

```
quarkus.datasource.devservices.enabled=true
quarkus.datasource.devservices.image-name=postgres:15
```

```
quarkus.s3.devservices.enabled=true
quarkus.aws.devservices.localstack.image-name=localstack/localstack:3.3.0
```

Не “паришься” по поводу локального/тестового запуска

```
quarkus.datasource.devservices.enabled=true
quarkus.datasource.devservices.image-name=postgres:15
```

```
quarkus.s3.devservices.enabled=true
quarkus.aws.devservices.localstack.image-name=localstack/localstack:3.3.0
```

```
quarkus.kafka.devservices.enabled=true
quarkus.kafka.devservices.image-name=confluentinc/cp-kafka:7.4.0
```

gRPC из коробки

Сразу **три** варианта:

1. старая Vert.x gRPC имплементация с **отдельным** gRPC сервером
2. новая Vert.x gRPC имплементаци **поверх текущего** HTTP сервера (тот же порт, без оверхеда)
3. xDS gRPC обертка над grpc-java с **отдельным** gRPC Netty сервером

gRPC из коробки

Сразу **три** варианта:

1. старая Vert.x gRPC имплементация с отдельным gRPC сервером
2. **новая Vert.x gRPC имплементаци поверх текущего HTTP сервера (тот же порт, без оверхеда)**
3. xDS gRPC обертка над grpc-java с отдельным gRPC Netty сервером

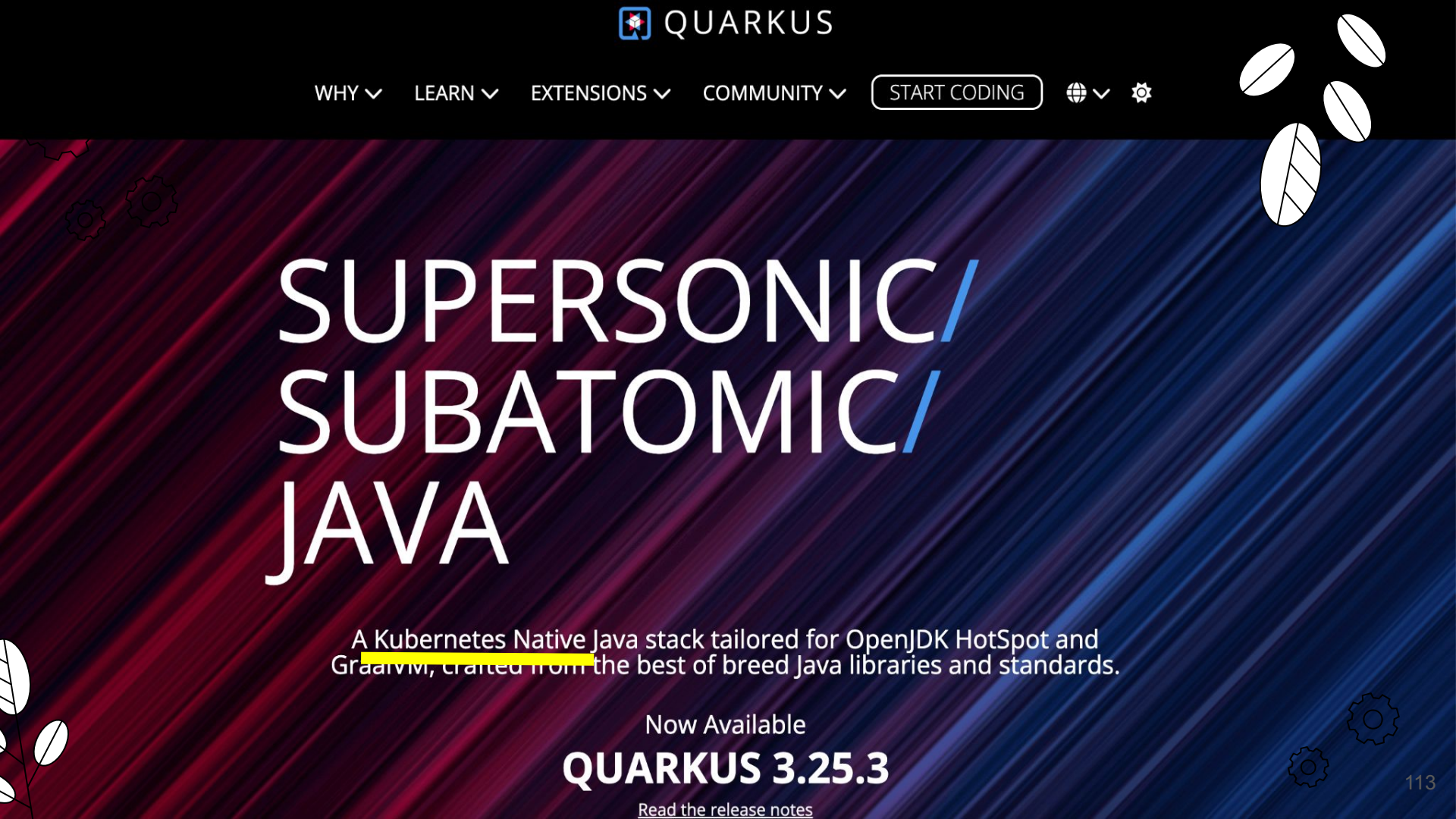
gRPC из коробки

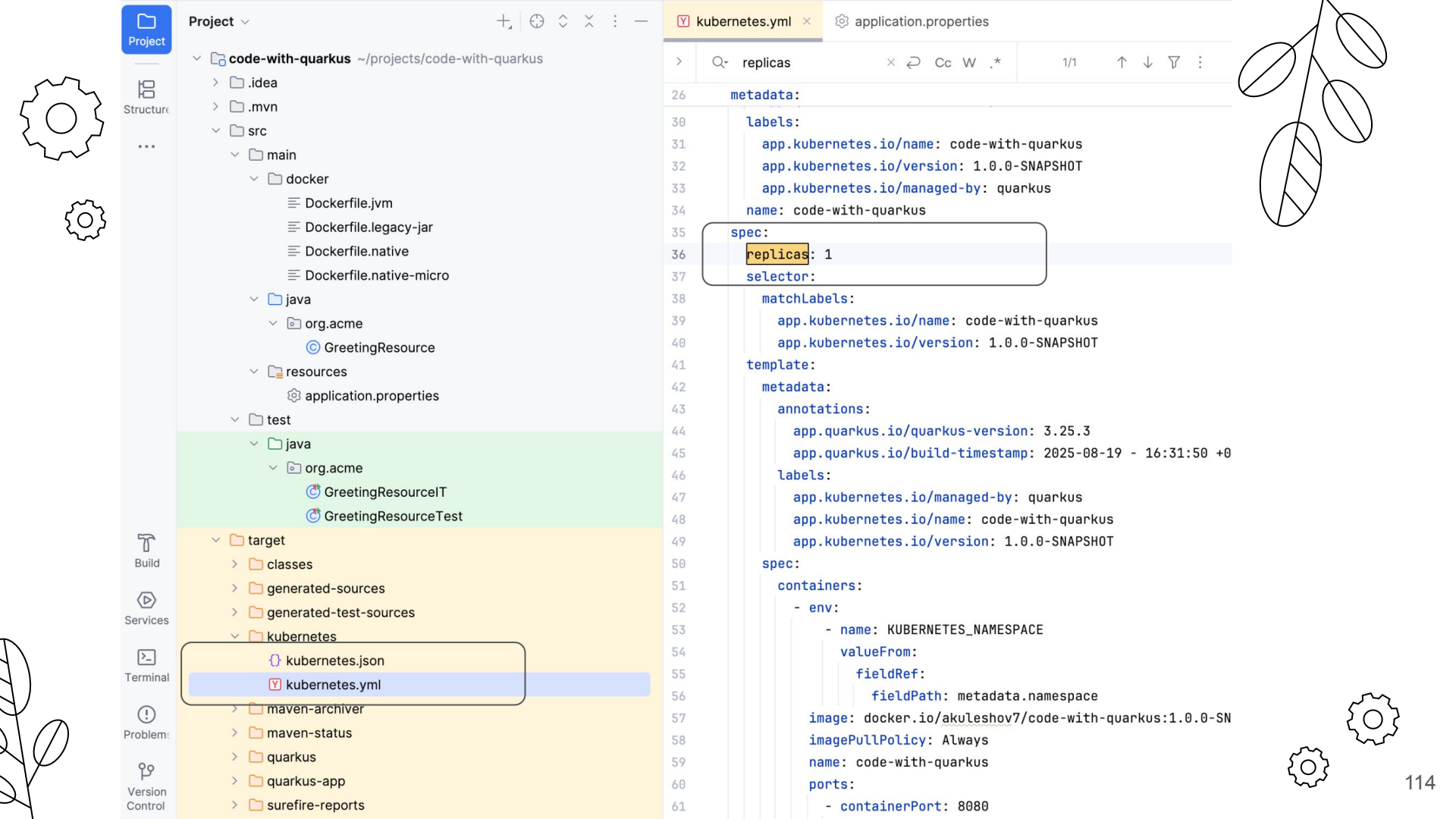
```
option java_package = "hello";

service GreeterService {
    rpc SayHello (HelloRequest) returns (HelloReply);
}
```

gRPC из коробки

```
import io.quarkus.grpc.GrpcService;
import hello.Greeter;

@GrpcService
public class HelloService implements GreeterService {

    @Override
    public Uni<HelloReply> sayHello(HelloRequest request) {
        return Uni.createFrom().item(() →
        HelloReply.newBuilder().setMessage(
            "Hello " + request.getName()).build()
        );
    }
}
```

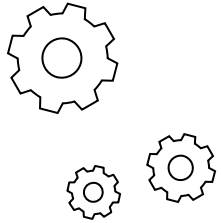

Так, ну ладно, разобрались.
А че они там писали про
Kubernetes Native?

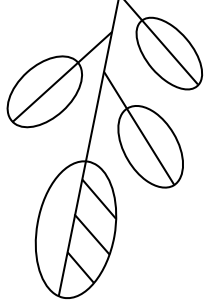

QUARKUS
WHY
LEARN
EXTENSIONS
COMMUNITY
START CODING
SUPERSONIC/
SUBATOMIC/
JAVA
A Kubernetes Native Java stack tailored for OpenJDK HotSpot and
GraalVM, crafted from the best of breed Java libraries and standards.
Now Available
QUARKUS 3.25.3
Read the release notes

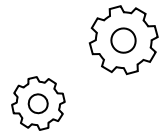

Project
Structure
Build
Services
Terminal
Problems
Version Control
Project
code-with-quarkus ~/projects/code-with-quarkus
.idea
.mvn
src
main
docker
Dockerfile.jvm
Dockerfile.legacy-jar
Dockerfile.native
Dockerfile.native-micro
java
org.acme
GreetingResource
resources
application.properties
test
java
org.acme
GreetingResourceIT
GreetingResourceTest
target
classes
generated-sources
generated-test-sources
kubernetes
kubernetes.json
kubernetes.yml
maven-archiver
maven-status
quarkus
quarkus-app
surefire-reports
kubernetes.yml
application.properties
replicas
1/1
26 metadata:
30 labels:
31 app.kubernetes.io/name: code-with-quarkus
32 app.kubernetes.io/version: 1.0.0-SNAPSHOT
33 app.kubernetes.io/managed-by: quarkus
34 name: code-with-quarkus
35 spec:
36 replicas: 1
37 selector:
38 matchLabels:
39 app.kubernetes.io/name: code-with-quarkus
40 app.kubernetes.io/version: 1.0.0-SNAPSHOT
41 template:
42 metadata:
43 annotations:
44 app.quarkus.io/quarkus-version: 3.25.3
45 app.quarkus.io/build-timestamp: 2025-08-19 - 16:31:50 +0
46 labels:
47 app.kubernetes.io/managed-by: quarkus
48 app.kubernetes.io/name: code-with-quarkus
49 app.kubernetes.io/version: 1.0.0-SNAPSHOT
50 spec:
51 containers:
52 - env:
53 - name: KUBERNETES_NAMESPACE
54 valueFrom:
55 fieldRef:
56 fieldPath: metadata.namespace
57 image: docker.io/akuleshov7/code-with-quarkus:1.0.0-SN
58 imagePullPolicy: Always
59 name: code-with-quarkus
60 ports:
61 - containerPort: 8080

application.properties

```
+ quarkus.kubernetes.replicas=7
```







application.properties

```
+ quarkus.kubernetes.replicas=7
```

```
spec:
 replicas: 7
...
```

application.properties

```
+  quarkus.kubernetes.replicas=7
```

```
+ dependency quarkus-smallrye-health
```

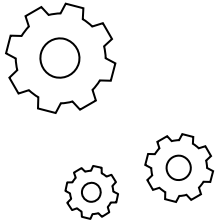
```
spec:
 replicas: 7
...
```

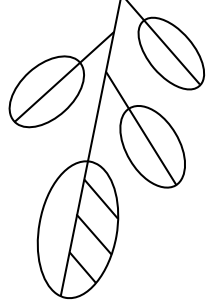
application.properties

```
+  quarkus.kubernetes.replicas=7
```

```
+ dependency quarkus-smallrye-health
```

```
spec:
 replicas: 7
...
livenessProbe:
...
 httpGet:
   path: /q/health/live
   port: 8080
   scheme: HTTP
...
readinessProbe:
 failureThreshold: 3
 httpGet:
   path: /q/health/ready
   port: 8080
   scheme: HTTP
```

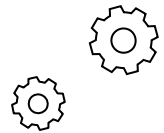


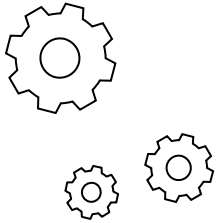


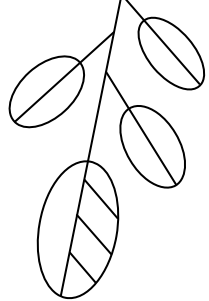
Кстати /q/ - это Management Interface

- /q/health/live
- /q/health/ready
- /q/health/started
- /q/metrics
- /q/openapi
- /q/swagger-ui

  ...

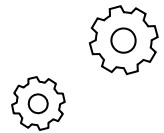


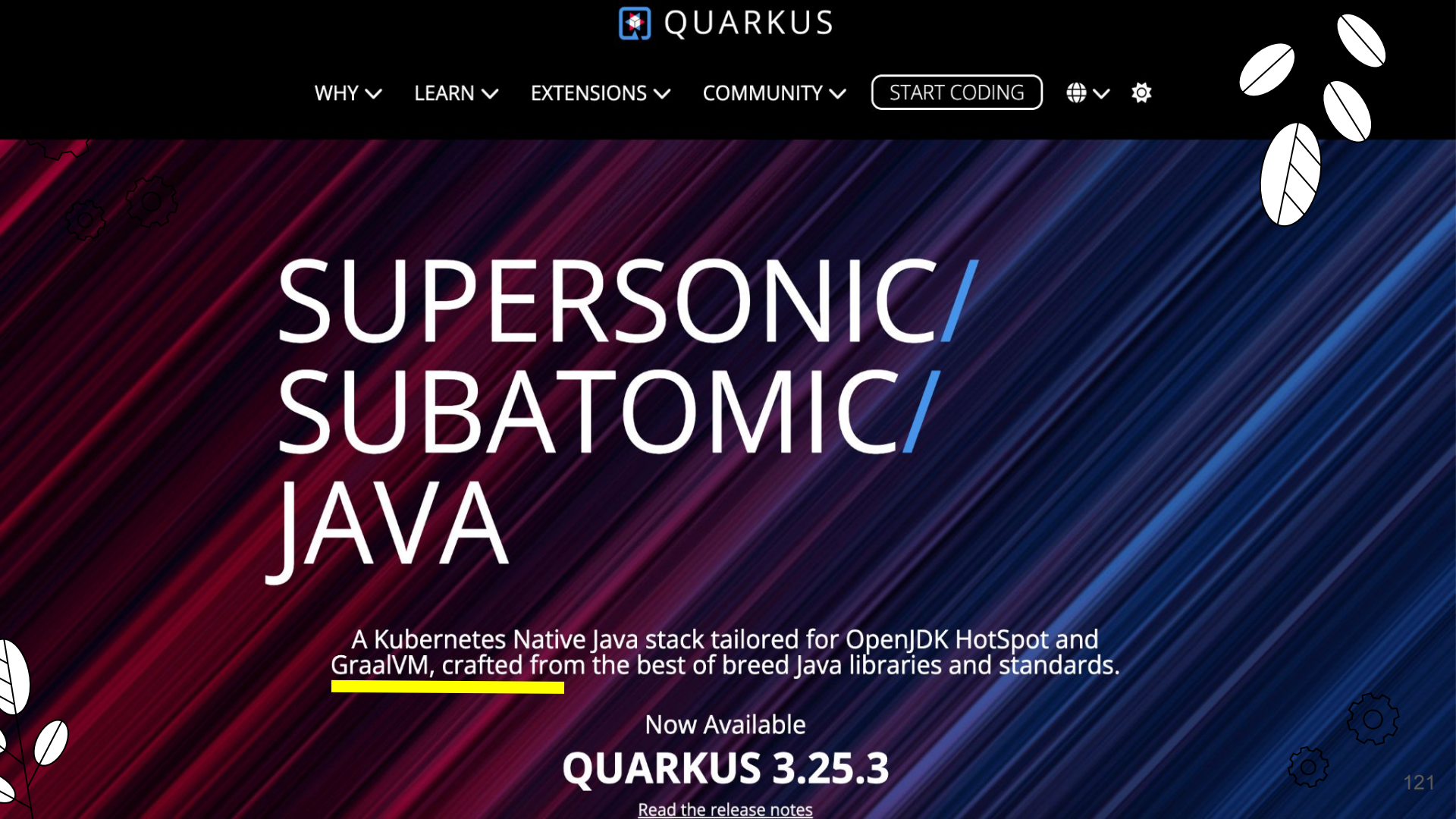


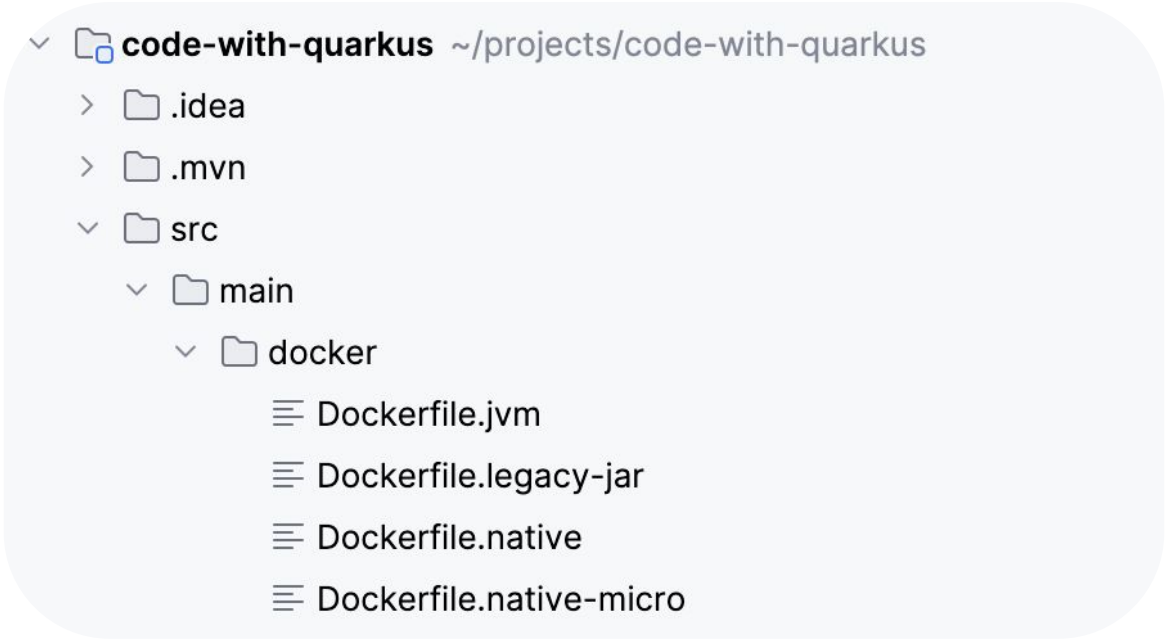


И /q/ легко вынести на отдельный HTTP сервер/порт, чтобы разделить от основной логики

```
quarkus.management.enabled=true

# на 0.0.0.0 по дефолту
quarkus.management.port=9004
```

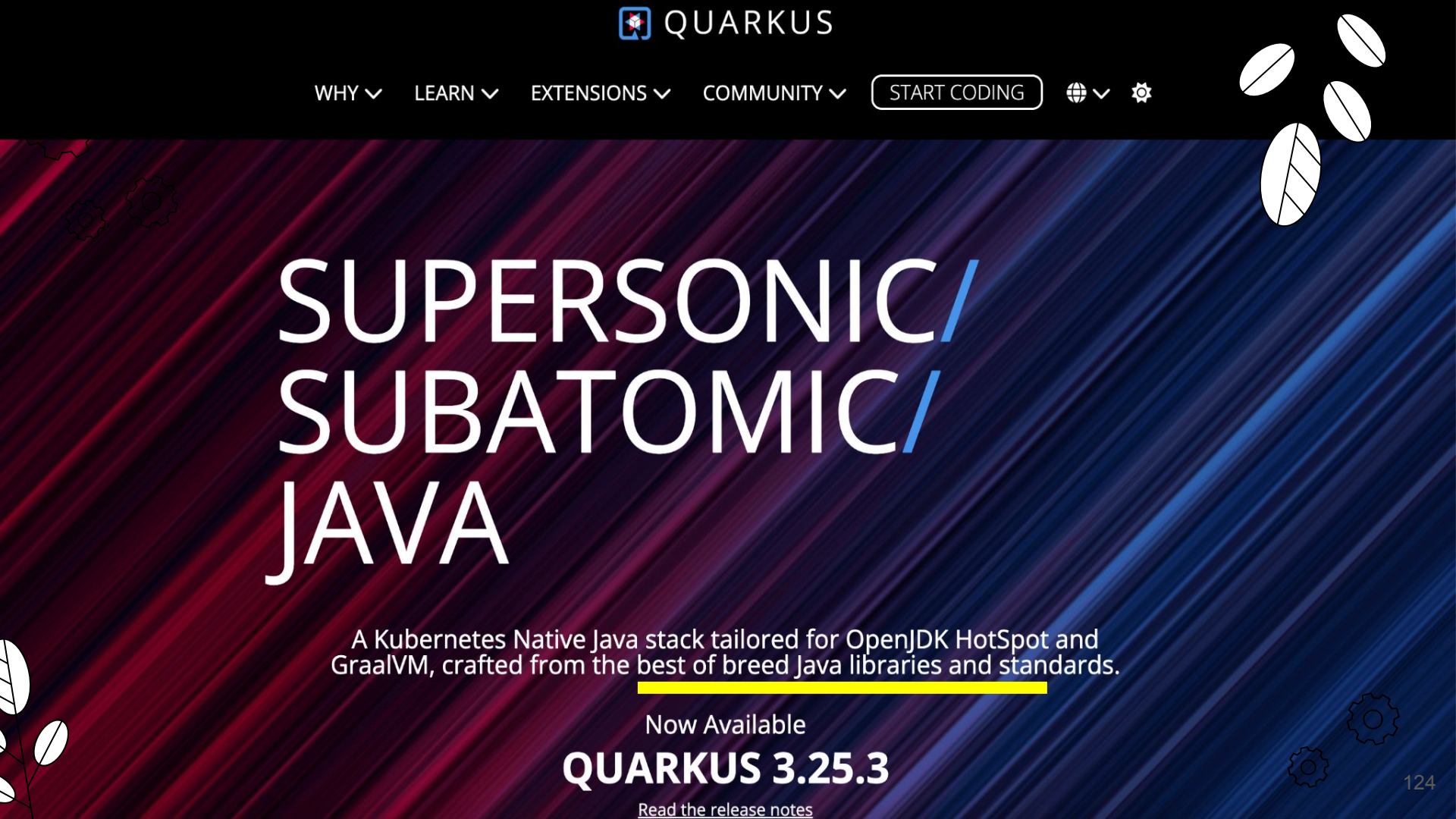



QUARKUS
WHY
LEARN
EXTENSIONS
COMMUNITY
START CODING
SUPERSONIC/
SUBATOMIC/
JAVA
A Kubernetes Native Java stack tailored for OpenJDK HotSpot and GraalVM, crafted from the best of breed Java libraries and standards.
Now Available
QUARKUS 3.25.3
Read the release notes

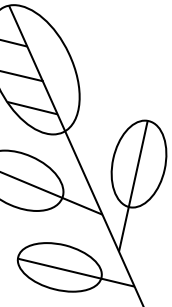



```
./mvnw package -Dnative -Dquarkus.native.container-build=true
```


Ладно, ясно, а в сортах чего они разбираются?

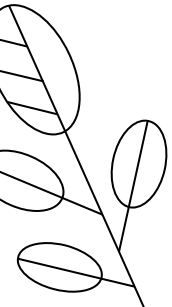

QUARKUS
WHY
LEARN
EXTENSIONS
COMMUNITY
START CODING
SUPERSONIC/
SUBATOMIC/
JAVA
A Kubernetes Native Java stack tailored for OpenJDK HotSpot and GraalVM, crafted from the best of breed Java libraries and standards.
Now Available
QUARKUS 3.25.3
Read the release notes

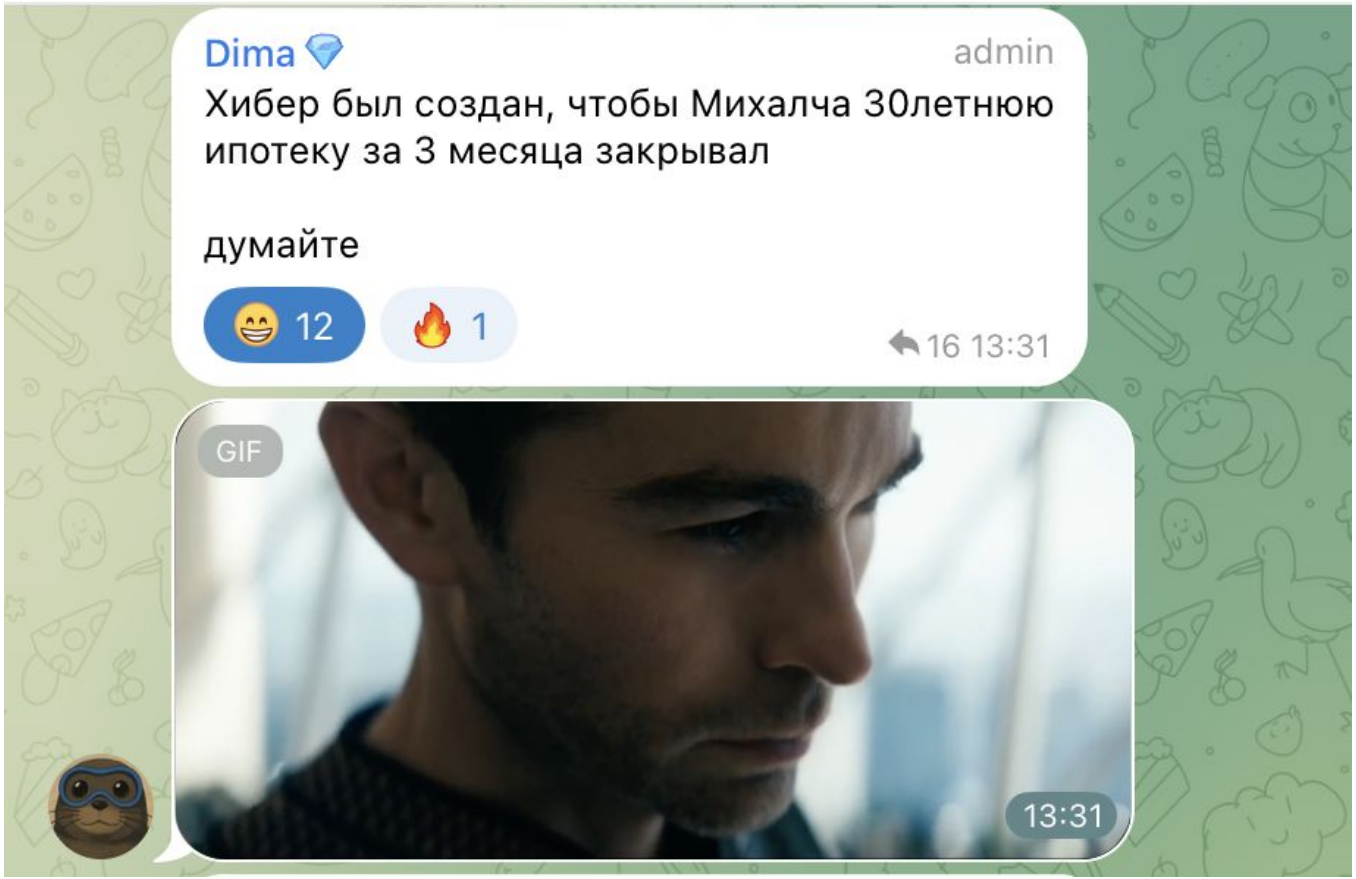
Ваши любимые ORM



Ваши любимые ORM

Урезанный Hibernate ORM с **Panache**




pro.jvm
6 136 members, 1903 online
Dima
admin
Хибер был создан, чтобы Михалча 30летнюю ипотеку за 3 месяца закрывал
думайте
12
1
16 13:31
GIF
13:31

Active Record Pattern

```
@Entity
public class Person extends PanacheEntity {
    public String name;
}
```

Active Record Pattern

```java
@Entity
public class Person extends PanacheEntity {
   public String name;
}
```

```java
Person p = new Person();
p.name = "Dima";
p.persist();
```

```java
List<Person> persons = Person.listAll();
Person byName = Person.find("name", "Dima").firstResult();
Person byName = Person.find("name =?1", "Dima").firstResult();
```

Active Record Pattern

```java
@Entity
public class Person extends PanacheEntity {
   public String name;
}
```

```java
Person p = new Person();
p.name = "Dima";
p.persist();
```

```java
List<Person> persons = Person.listAll();
Person byName = Person.find("name", "Dima").firstResult();
Person byName = Person.find("name =?1", "Dima").firstResult();
```

Repository Pattern

```
@ApplicationScoped
public class PersonRepository
    implements PanacheRepository<Person> {
   ...
}

repo.persist(new Person("Dima", 30));
List<Person> people = repo.listAll();
```

Datasource

```
# Default
quarkus.datasource.db-kind=postgresql
quarkus.datasource.username=user1
quarkus.datasource.password=pass1
quarkus.datasource.jdbc.url=jdbc:postgresql://

# Named datasource: orders
quarkus.datasource.orders.db-kind=mysql
quarkus.datasource.orders.username=user2
quarkus.datasource.orders.password=pass2
quarkus.datasource.orders.jdbc.url=jdbc:mysql://
```

Datasource

```
# Default
quarkus.datasource.db-kind=postgresql
quarkus.datasource.username=user1
quarkus.datasource.password=pass1
quarkus.datasource.jdbc.url=jdbc:postgresql://

# Named datasource: orders
quarkus.datasource.orders.db-kind=mysql
quarkus.datasource.orders.username=user2
quarkus.datasource.orders.password=pass2
quarkus.datasource.orders.jdbc.url=jdbc:mysql://
```

Пул JDBC коннекшенов через Agroal (свой Hikari)

Есть и “нестандартная” Native Query

```
@Entity
@NamedNativeQuery(
    name = "Person.getByAge",
    query = "SELECT * FROM person WHERE age > :age",
    resultClass = Person.class
    )
public class Person extends PanacheEntity {
        public String name;
        public int age;
    }
```

Что еще?

Что еще?

@Transactional

JBoss / Narayana Transaction Manager

Что еще?

@Transactional

JBoss / Narayana Transaction Manager

hibernate-**reactive**-panache

- ChecklistForNewProjects
- DocumentingYourExtension
- Home
- Infrastructure for Quarkus 3.x: Manual upgrade
- Migrating to the Sonatype Portal API using JReleaser
- Release
- NamingConventions
- Infrastructure for Quarkus 3.x

https://hub.quarkiverse.io/
https://github.com/quarkiverse

Welcome to the Quarkiverse Hub!

Quarkus is a Kubernetes Native Java stack tailored for the OpenJDK HotSpot and GraalVM, crafted from the best of breed Java libraries and standards, as well as an ever growing ecosystem of its extensions. At the beginning we were accepting all the contributed extensions in the core Quarkus repository. Eventually though it grew too large and lead to a maintenance overhead. In addition to that, in some cases it's simply not always sensible to include an extension into the main Quarkus repository. For example, an extension might want to have a release cadence which is independent of the core repository's lifecycle.

Quarkiverse is our solution to create a "home" for such extensions.

What is Quarkiverse

The Quarkiverse GitHub organization provides repository hosting (including build, CI and release publishing setup) for Quarkus extension projects contributed by the community.

Quarkus extensions hosted in the Quarkiverse organization can easily be included into the Quarkus extensions catalog available on code.quarkus.io, extensions.quarkus.io, and the Quarkus command line tools (such as `mvn quarkus:list-extensions`, `gradle listExtensions`). To stay listed, the only requirement is that the extension keeps functioning, stays up-to-date and cause no harm.

Why Quarkiverse

In the early days and actually up until recently, Quarkus extensions contributed by the community members (including the core Quarkus t

IntelliJ IDEA CE

welcomed in the main


quarkus-jooq
Public
Quarkus Jooq Extension
quarkus-extension
Java • Apache License 2.0 • 19 • 77 • 20 • 2 • Updated last week

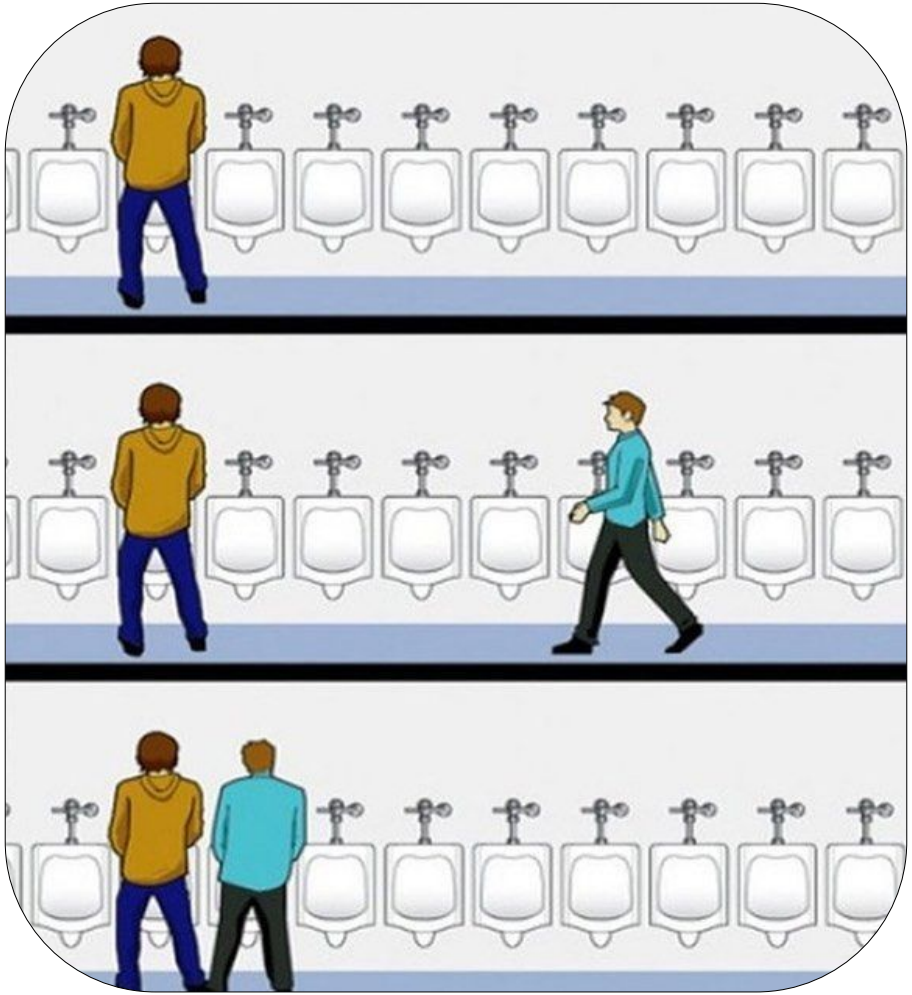

quarkus-mcp-servers
Public
Model Context Protocol Servers in Quarkus
mcp
quarkus-app
Java • Apache License 2.0 • 41 • 163 • 16 • 6 • Updated 3 weeks ago


quarkus-scala3
Public
Quarkus Extension to support Scala 3
quarkus-extension
Java • Apache License 2.0 • 6 • 59 • 2 • 9 • Updated on Jan 9


quarkus-amazon-services
Public
Quarkus Amazon Services extensions
quarkus-extension
Java • Apache License 2.0 • 62 • 48 • 4 • 4 • Updated 1 hour ago


quarkus-neo4j
Public
Quarkus Neo4j extension
neo4j
graph
graph-database
quarkus-extension
Java • Apache License 2.0 • 7 • 23 • 1 • 2 • Updated last week

Выводы




Как там, переехал
на третий спринг бут?

Выводы

Quarkus - это:

Выводы

Quarkus - это:

- скорость

Выводы

Quarkus - это:

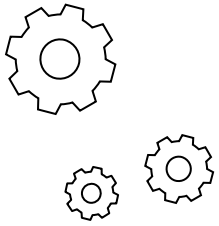
- скорость
- отдельная экосистема

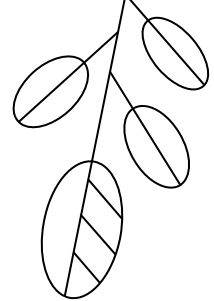
Выводы

Quarkus - это:

- скорость
- отдельная экосистема
- делалось корпорацией для корпораций

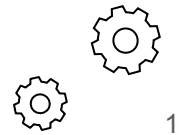
Выводы



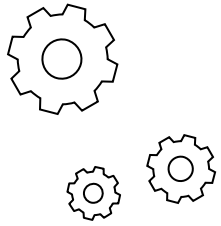


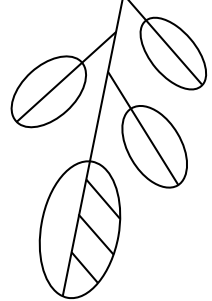
Quarkus - это:

- скорость
- отдельная экосистема
- делалось корпорацией для корпораций
- шероховатости, до сих пор полируется



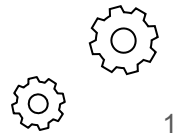
Выводы





Quarkus - это:

- скорость
- отдельная экосистема
- делалось корпорацией для корпораций
- шероховатости, до сих пор полируется
- быстро и безболезненно для POC/MVP

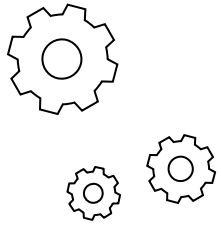
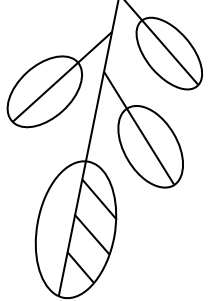


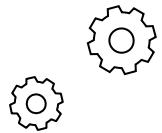
Спасибо!

Q&A

Кулешов

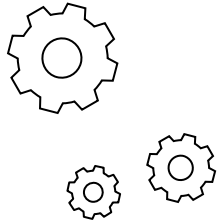
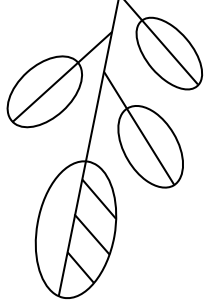
Андрей, с чем ты неожиданным сталкивался?

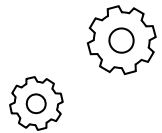


1

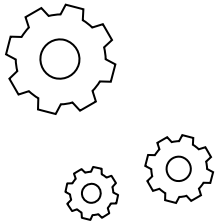
JBoss Logging

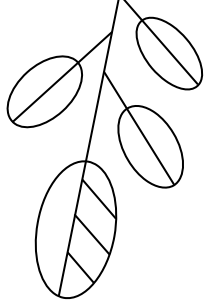



2

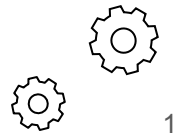



Столкновение с метриками





By default, the metrics are exported using the Prometheus format ***application/openmetrics-text****, you can revert to the former format by specifying the Accept request header to text/plain (curl -H "Accept: text/plain" localhost:8080/q/metrics/).*



Configurable metrics format #49152

New issue

Open #49581

axidex opened 3 weeks ago

Description

Some custom metrics exporters sidecars don't have ability to add custom headers to requests. It would be great if this could be added to the quarkus.micrometer config.

> By default, the metrics are exported using the Prometheus format application/openmetrics-text, you can revert to the former format by specifying the Accept request header to text/plain (curl -H "Accept: text/plain" localhost:8080/q/metrics/).

Implementation ideas

I would suggest something like this:

```
quarkus.micrometer.export.prometheus.format=plain
quarkus.micrometer.export.prometheus.format=openmetrics
```

4

axidex added kind/enhancement 3 weeks ago

quarkus-bot added area/config area/metrics area/smallrye 3 weeks ago

Assignees

No one assigned

Labels

area/metrics kind/enhancement

Type

No type

Projects

No projects

Milestone

No milestone

Relationships

None yet

Development

Code with agent mode

Format property added in micrometer prometheus config

quarkusio/quarkus

Open **Format property added in micrometer prometheus config** #49581

axidex wants to merge 4 commits into `quarkusio:main` from `axidex:feature/micrometer-prom…`

Configuration

```
# Use legacy Prometheus text format by default
quarkus.micrometer.export.prometheus.format=plain

# Use OpenMetrics format by default (default behavior)
quarkus.micrometer.export.prometheus.format=openmetrics
```

Behavior

- When no Accept header is provided, the configured format is used
- When Accept header is provided (e.g., text/plain or application/openmetrics-text), it takes precedence over configuration
- This allows custom metrics exporters/sidecars that don't support custom Accept headers to work with the plain text format

Issues

- Fixes Configurable metrics format #49152

Format property added in micrometer prometheus config — df3c38f

quarkus-bot bot added the area/metrics label 5 days ago

quarkus-bot bot commented 5 days ago

/cc **@brunobat** (micrometer), **@ebullient** (micrometer)

axidex marked this pull request as ready for review 5 days ago

gastaldi reviewed 3 days ago — View reviewed changes

Labels

area/metrics

Projects

None yet

Milestone

No milestone

Development

Successfully merging this pull request may close these issues.

Configurable metrics format

Notifications — Customize

Subscribe

You're not receiving notifications from this thread.

2 participants

Open **Format property added in micrometer prometheus config** #49581

axidex wants to merge 4 commits into `quarkusio:main` from `axidex:feature/micrometer-prom…`

Configuration

```
# Use legacy Prometheus text format by default
quarkus.micrometer.export.prometheus.format=plain

# Use OpenMetrics format by default (default behavior)
quarkus.micrometer.ex
```

Behavior

- When no Accept head
- When Accept header
- This allows custom m
  format

over configuration
the plain text

Issues

- Fixes Configurable

Format property df3c38f

quarkus-bot bot

quarkus-bot bot comm

/cc @brunobat (microme

axidex marked this pull request as ready for review 5 days ago

gastaldi reviewed 3 days ago View reviewed changes

Labels

area/metrics

Projects

None yet

Milestone

No milestone

Development

Successfully merging this pull request may close these issues.

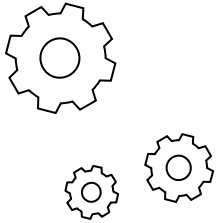
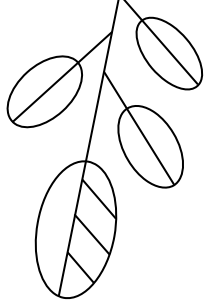
Configurable metrics format

Notifications Customize

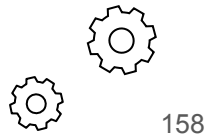
Subscribe

You're not receiving notifications from this thread.

2 participants

3

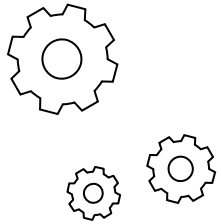
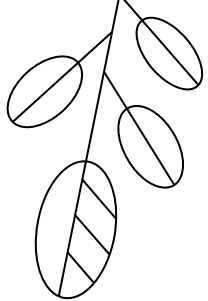
Неожиданные проблемы в неожиданных местах

```
@GET
@Path("/{organization}/{repo}/{commit}")
fun getResults(): Response {
    val list = listOf<B>(B("1"), B("2"))
    return Response.ok(list).build()
}

@Serializable
data class B(
    val b: String
)
```

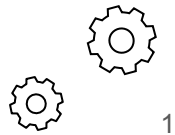
Неожиданные проблемы в неожиданных местах

```
@GET
@Path("/{organization}/{repo}/{commit}")
fun getResults(): Response {
    val list = listOf<B>(B("1"), B("2"))
    return Response.ok(list).build()
}

@Serializable
data class B(
    val b: String
)
```

```
SerializationException: Serializer for class 'ArrayList' is not found.
```

4

Is automatic validation of Protobuf messages in place? #42802

Unanswered **preslavrachev** asked this question in **Q&A**

preslavrachev on Aug 27, 2024

Take the following Protobufs spec:

```
message SomeMessageRequest {
    string name = 1;
    optional string label = 2;
}
```

I also have a Grpc service that is suposed to accept requests of that type. A quick demo allowed me to send and empty message to that service, and it still hit the endpoint. I was expecting an error, because the `name` field is not marked as `optional`, but that wasn't the case. The response from the server was a clear 200 OK.

What is the way to validate incoming Protobuf messages in Quarkus, other than field-by-field checking?

↑ 2

1 comment

Oldest | Newest | Top

cescoffier on Sep 2, 2024 Maintainer

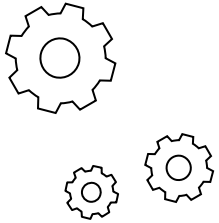
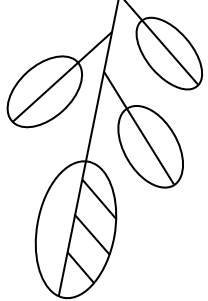
No, there is no automatic validation. We follow the same default behavior of gRPC Java. If that's not the case anymore, please open an issue.

↑ 1

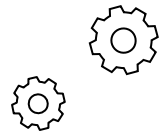
0 replies

Write a reply

Генерация из proto, валидации и плагины. Голая и куцая имплементация

5

August 29, 2025 #release

Quarkus 3.26.1, 3.20.2.2 and 3.15.6.2 - Emergency releases

By Guillaume Smet

Today, we released Quarkus 3.26.1, 3.20.2.2 and 3.15.6.2 to fix an important regression introduced in Vert.x 4.5.18.

We recommend upgrading to these releases as soon as possible if you are using a Quarkus version using Vert.x 4.5.18 (i.e. 3.26.0, 3.25.4, 3.20.2.1, and 3.15.6.1).

Quarkus 3.26.1 also contains some other unrelated fixes.

Спасибо!

Q&A

Обратная связь

Мой канал