JAVA FAKTURA

S02: Stranger Spring

E02: Spring re-Booted

Adam Król

29/10/2019

Spring Boot

JAVA FAKTURA

Spring Boot makes it easy to create stand-alone, production-grade Spring based Applications that you can "just run".

@https://spring.io/projects/spring-boot

Spring Boot Package

JAVA FAKTURA

- **Problem: Jak dostarczyć aplikację?**
  - Zależności
  - Uruchomienie
- **Rozwiązanie: Fat/Uber Jar**

Spring Boot Package vs Fat Jar:

- Dodając Spring Boot plugin (Gradle, Maven, Ant) wszystko dzieje się automatycznie
- Lepsze oddzielenie klas naszej aplikacji od zależności - dzięki spring-boot-loader

Spring Boot Loader

JAVA FAKTURA

```
src
|       + — main
|       |   + — java
|       |   |   └— io
|       |   |       └— javafaktura
|       |   |           └— app
|       |   |               └— ScrapperApplication.java
```

```
+—io
|   └— javafaktura
|       └— app
|           └— ScrapperApplication.class
└— META-INF
    └— MANIFEST.MF
```

Spring Boot Loader

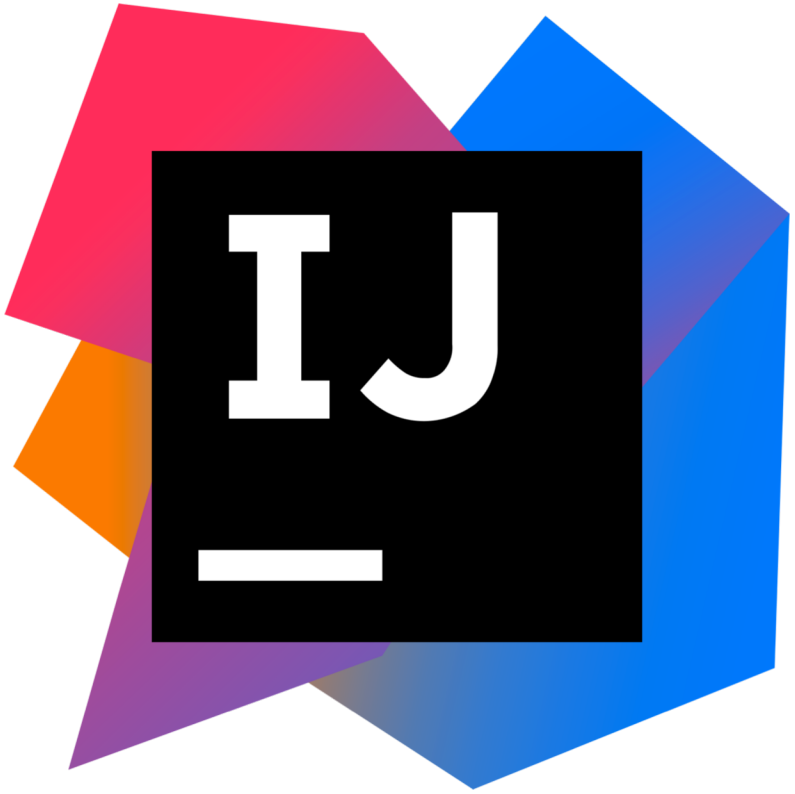
JAVA FAKTURA

```
src
|       + — main
|       |   + — java
|       |   |   └— io
|       |   |       └— javafaktura
|       |   |           └— app
|       |   |               └— ScrapperApplication.java
```

```
+—org
|   └— springframework
|       └— boot
|           └— JarLauncher.class
+—META-INF
|   └— MANIFEST.MF
└— BOOT-INF
    +— classes
    |   └— io
    |       └— javafaktura
    |           └— app
    |               └— ScrapperApplication.class
    └— lib
        └— jsoup.jar
```

Spring Boot Jar

JAVA FAKTURA



@SpringBootApplication

JAVA FAKTURA

- Skanuje wszystkie beany w głównym pakiecie naszej aplikacji (@ComponentScan)
- Dodaje @SpringBootConfiguration – główna konfiguracja całej aplikacji
- Włącza automatyczne skanowanie wszystkich autokonfiguratorów (@EnableAutoConfiguration)

SpringApplication

JAVA FAKTURA

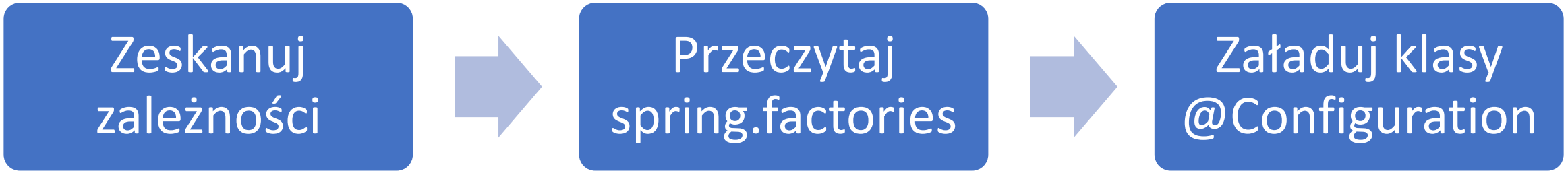
```
@SpringBootApplication
public class ApplicationMain {

   public static void main(String[] args) {
            SpringApplication.run(ApplicationMain.class, args);
   }
}
```

Co SpringApplication.run robi dla nas przy starcie?

- Buduje kontekst z wykorzystaniem AutoConfiguration oraz go odświeża
- Analizuje przekazane args i automatycznie zamienia to na properties Springa
- Startuje wszystkie *CommandLineRunnery*

@EnableAutoConfiguration phase

JAVA FAKTURA




```
org.springframework.boot.autoconfigure.EnableAutoConfiguration=\
org.springframework.boot.actuate.autoconfigure.amqp.RabbitHealthIndicatorAutoConfiguration,\
org.springframework.boot.actuate.autoconfigure.audit.AuditAutoConfiguration,\
```

WAŻNE!
Cała ta faza odbywa się zanim nasze Beany powstaną.

Spring Boot Starters – Pozwól, że to skonfiguruję

JAVA FAKTURA

Bez Startera:

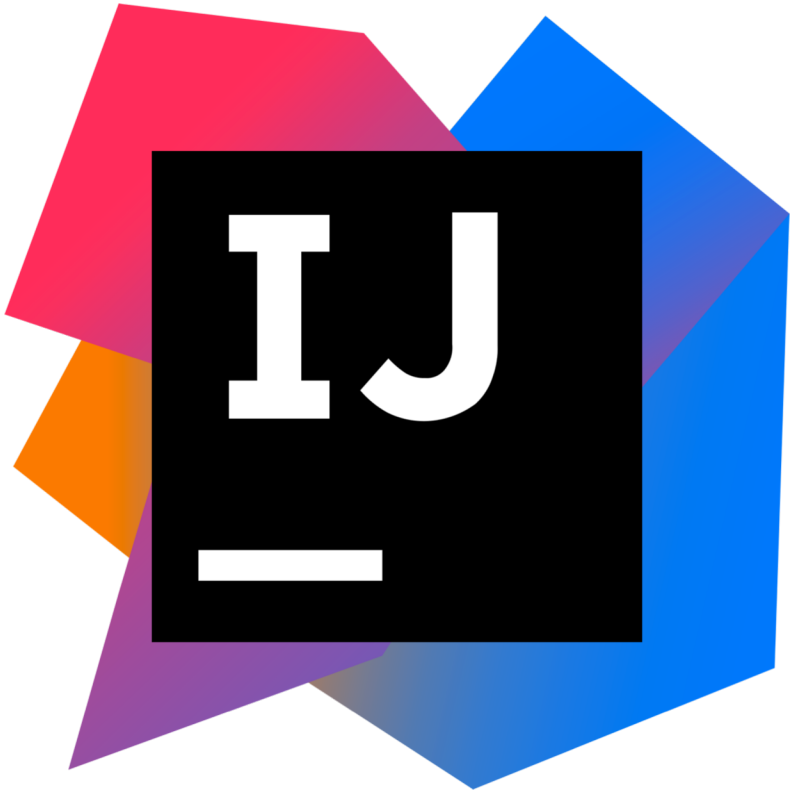
- Dodajemy zależność np. spring-data-jpa
- Włączamy ten moduł – dodajemy @EnableJpaRepositories
- Konfigurjemy DataSource, EntityManagerFactory
- Używamy!

Ze Starterem:

- Dodajemy zależność: spring-boot-starter-data-jpa
- Używamy!

Aplikacja Spring Boot ze starterem

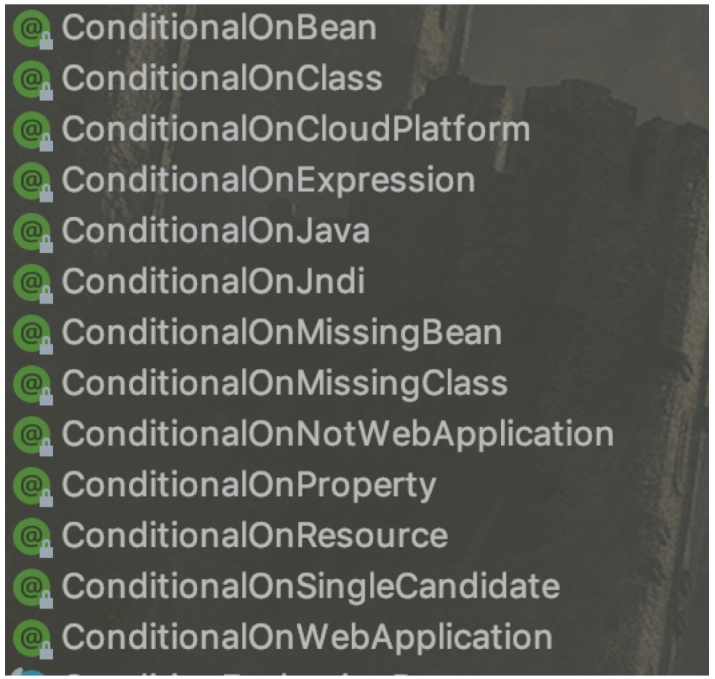
JAVA FAKTURA



Czy wszystko trzeba ładować? - Conditional

JAVA FAKTURA

Zeskanuj zależności → Przeczytaj spring.factories → Sprawdź warunki dla tej konfiguracji → Załaduj klasy @Configuration




```
@Conditional(OnJavaCondition.class)
public @interface ConditionalOnJava {
  ...
}

class OnJavaCondition extends SpringBootCondition {
        @Override
        public ConditionOutcome getMatchOutcome(ConditionContext context,
                AnnotatedTypeMetadata metadata) {
                ...
                return getMatchOutcome(range, JVM_VERSION, version);
        }
}
```

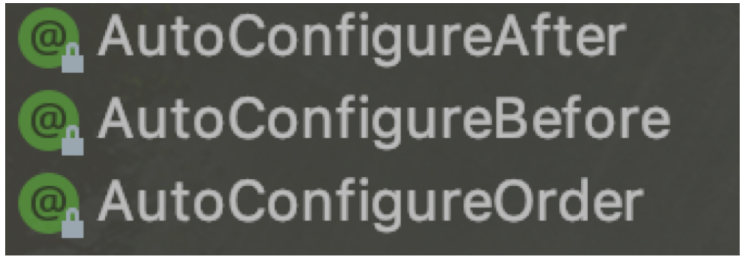
Jak nad tym zapanować? – Kolejność konfiguracji

JAVA FAKTURA

```
@Configuration
public class AConfig {

    @Bean
    AnotherDep anotherDependency(Dep dep) {
        return new AnotherDep(dep);
    }

}
```

```
@Configuration
public class BConfig {

    @Bean
    Dependency dependency() {
        return new Dependency();
    }
}
```

```
@AutoConfigureAfter(BConfig.class)
```



Spring Boot Actuator

JAVA FAKTURA

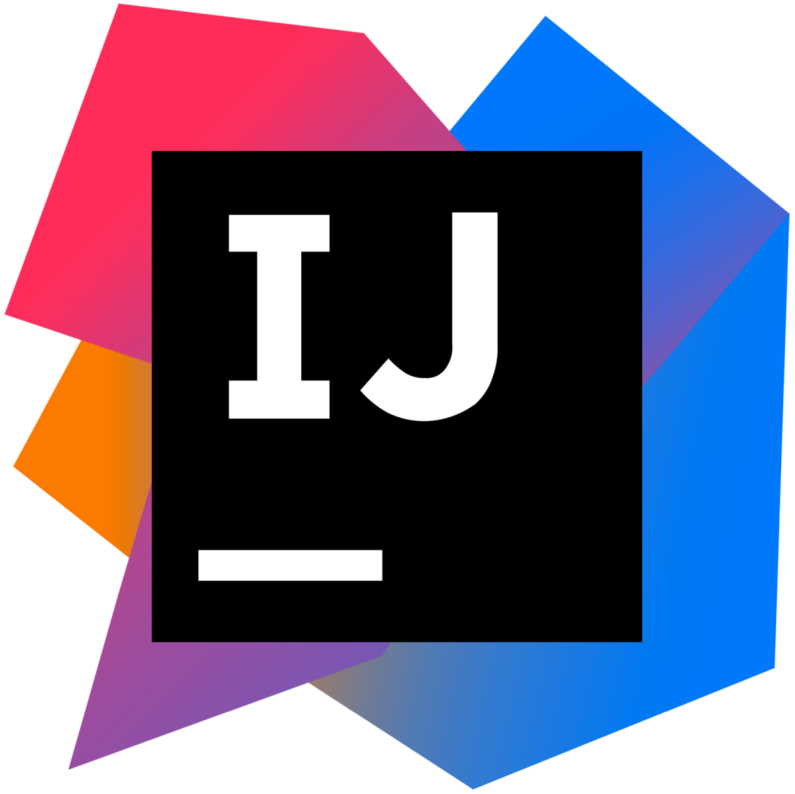
/actuator/health – status naszej aplikacji. Zazwyczaj zobaczymy tam „status: UP”

/actuator/info – informacje dotyczące działającej aplikacji. Możemy tutaj znaleźć wersję, git commit itp. (Wszystko bazuje na pliku (META-INF/build-info.properties)

/actuator/prometheus – Metryki aplikacji (biznesowe i aplikacyjne) wystawione w formacie dla Prometheusa

Spring Boot Actuator + Startery

JAVA FAKTURA



Spring Boot Failure Analyzer

JAVA FAKTURA

```
Caused by: org.springframework.beans.BeanInstantiationException: Failed to instantiate [io.javafaktura.spring.boot.allegro.AllegroScrapper]: Constructor threw exception; nested exception
  is io.javafaktura.spring.boot.allegro.UnableToSetupAllegroException: Allegro lokalnie is not available
    at org.springframework.beans.BeanUtils.instantiateClass(BeanUtils.java:213) ~[spring-beans-5.2.0.RELEASE.jar:5.2.0.RELEASE]
    at org.springframework.beans.factory.support.SimpleInstantiationStrategy.instantiate(SimpleInstantiationStrategy.java:117) ~[spring-beans-5.2.0.RELEASE.jar:5.2.0.RELEASE]
    at org.springframework.beans.factory.support.ConstructorResolver.instantiate(ConstructorResolver.java:308) ~[spring-beans-5.2.0.RELEASE.jar:5.2.0.RELEASE]
    ... 33 common frames omitted
Caused by: io.javafaktura.spring.boot.allegro.UnableToSetupAllegroException: Allegro lokalnie is not available
    at io.javafaktura.spring.boot.allegro.AllegroScrapper.checkAllegro(AllegroScrapper.java:45) ~[main/:na]
    at io.javafaktura.spring.boot.allegro.AllegroScrapper.<init>(AllegroScrapper.java:36) ~[main/:na] <4 internal calls>
    at org.springframework.beans.BeanUtils.instantiateClass(BeanUtils.java:200) ~[spring-beans-5.2.0.RELEASE.jar:5.2.0.RELEASE]
    ... 35 common frames omitted
Caused by: java.net.UnknownHostException: allegrolokalnie.pl
    at java.base/java.net.AbstractPlainSocketImpl.connect(AbstractPlainSocketImpl.java:220) ~[na:na]
    at java.base/java.net.SocksSocketImpl.connect(SocksSocketImpl.java:403) ~[na:na]
    at java.base/java.net.Socket.connect(Socket.java:591) ~[na:na]
    at io.javafaktura.spring.boot.allegro.AllegroScrapper.checkAllegro(AllegroScrapper.java:42) ~[main/:na]
    ... 41 common frames omitted
```

```
***************************
APPLICATION FAILED TO START
***************************

Description:

Allegro could not be initialized.

Action:

Maybe turn on internet.
```

```
public class AllegroExceptionFailureAnalyzer
                extends AbstractFailureAnalyzer<Exception> {
}
```

- Lepsze wyjaśnienie błędów
- Możliwość zasugerowania rozwiązania

Cykl życia Spring Boot

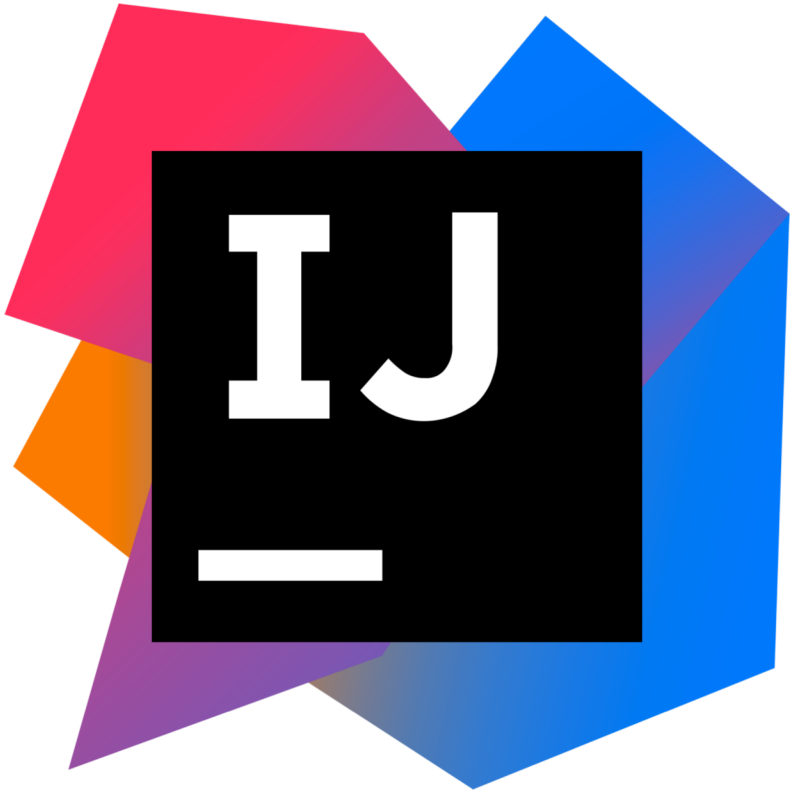
JAVA FAKTURA

- ApplicationStartingEvent
- ApplicationEnvironmentPreparedEvent
- ApplicationPreparedEvent
- ApplicationContextInitializedEvent
- ApplicationStartedEvent
- ApplicationReadyEvent

- ApplicationFailedEvent

Spring Boot Failure Analyzer

JAVA FAKTURA

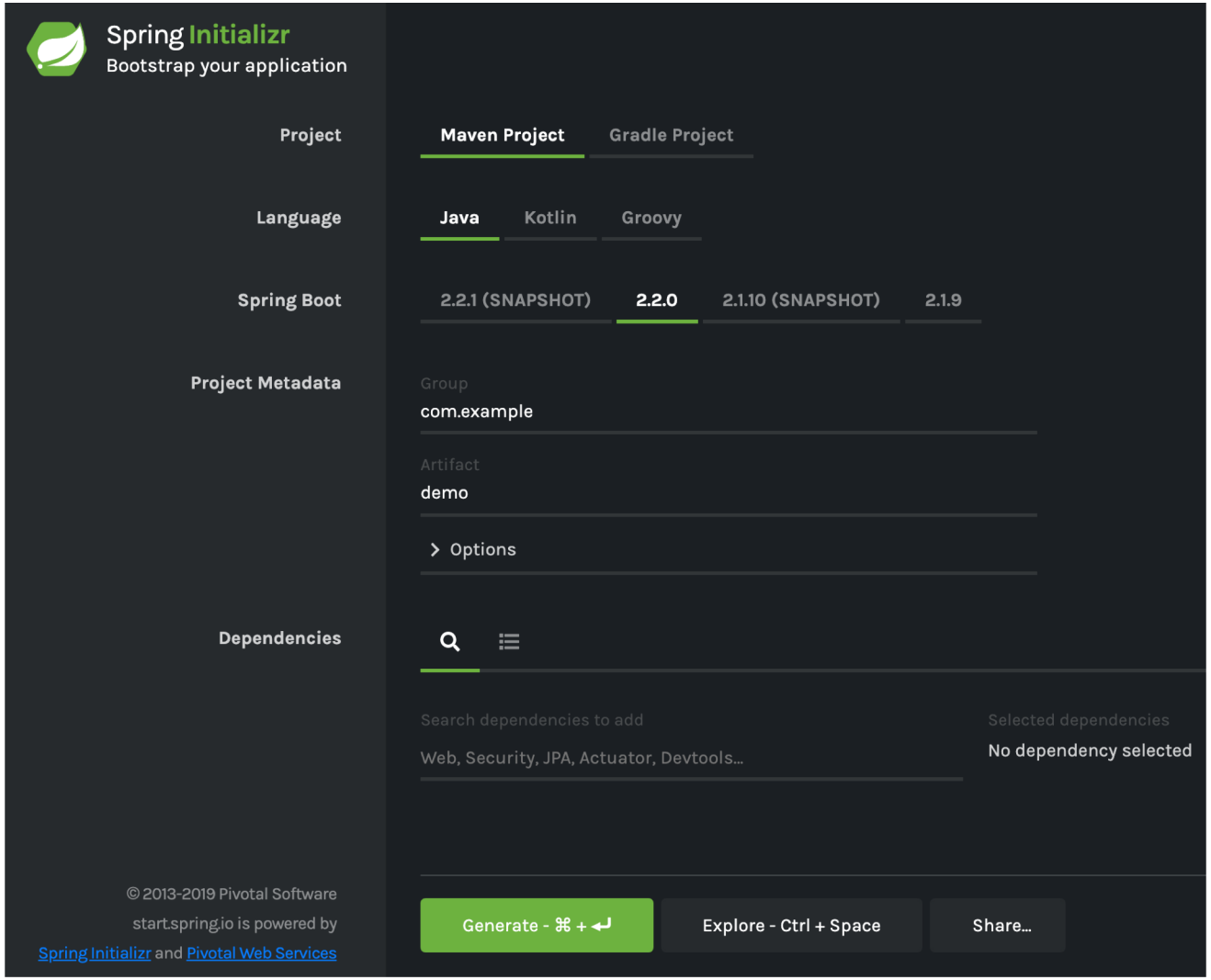


Spring Boot Initializr

JAVA FAKTURA

Spring Boot Initializr

JAVA FAKTURA



Spring Boot - Podsumowanie

JAVA FAKTURA

Dziękuję za uwagę

JAVA FAKTURA

https://tinyurl.com/jfs02e02survey