

Uputstvo za integraciju

Potrebno je kreirati tri Spring Boot projekta. Svaki od projekata ima svoju funkciju:

1. Model – smeštamo model aplikacije
2. Service – služi za servisni i kontroler sloj
3. Kjar – sva pravila (drl i drt fajlovi)

Vaše Spring Boot projekte možete da generišete na sajtu na [linku](#).

Service projekat



Project
☐ Gradle - Groovy ☐ Gradle - Kotlin ☒ Maven

Language
☒ Java ☐ Kotlin ☐ Groovy

Spring Boot
☐ 3.1.0 (SNAPSHOT) ☐ 3.1.0 (M1) ☐ 3.0.4 (SNAPSHOT) ☐ 3.0.3
☐ 2.7.10 (SNAPSHOT) ☒ 2.7.9

Project Metadata

Group

Artifact

Name

Description

Package name

Packaging ☒ Jar ☐ War

Java ☐ 19 ☐ 17 ☒ 11 ☐ 8

Dependencies

ADD DEPENDENCIES... CTRL + B

Spring Web WEB

Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Model projekat



Project
☐ Gradle - Groovy ☐ Gradle - Kotlin ☒ Maven

Language
☒ Java ☐ Kotlin ☐ Groovy

Spring Boot
☐ 3.1.0 (SNAPSHOT) ☐ 3.1.0 (M1) ☐ 3.0.4 (SNAPSHOT) ☐ 3.0.3
☐ 2.7.10 (SNAPSHOT) ☒ 2.7.9

Project Metadata

Group

Artifact

Name

Description

Package name

Packaging ☒ Jar ☐ War

Java ☐ 19 ☐ 17 ☒ 11 ☐ 8

Dependencies

ADD DEPENDENCIES... CTRL + B

No dependency selected

Kjar projekat

The image shows the Spring Initializr web interface. It has three main sections: Project, Language, and Dependencies. In the Project section, 'Maven' is selected. In the Language section, 'Java' is selected. In the Spring Boot section, '2.7.10 (SNAPSHOT)' is selected. The Project Metadata section contains fields for Group (com.ftn.sbnz), Artifact (kjar), Name (kjar), Description (Demo project for Spring Boot), Package name (com.ftn.sbnz.kjar), Packaging (Jar), and Java version (11). The Dependencies section is empty with a button to add dependencies.

Project
☐ Gradle - Groovy ☐ Gradle - Kotlin ☒ **Java** ☐ Kotlin ☐ Groovy

Language
☒ **Java** ☐ Kotlin ☐ Groovy

Spring Boot
☐ 3.1.0 (SNAPSHOT) ☐ 3.1.0 (M1) ☐ 3.0.4 (SNAPSHOT) ☐ 3.0.3 ☒ **2.7.10 (SNAPSHOT)**

Project Metadata
Group:
Artifact:
Name:
Description:
Package name:
Packaging: ☒ **Jar** ☐ War
Java: ☐ 19 ☐ 17 ☒ **11** ☐ 8

Dependencies
ADD DEPENDENCIES... CTRL + B
No dependency selected

Zatim raspakujete projekat u jedan folder. Sledeći korak je da sredimo postojeće pakete, dodamo zavisnosti i povežemo međusobno projekte.

Prvo otvorimo folder u kome smo raspakovali projekte u VS Code-u. Potrebno je da instalirate ekstenzije [Extension Pack for Java](#) i [Drools](#).

Pošto želimo da nam *service* bude glavni projekat koji ćemo pokretati, u *model* i *kjar* projektima možemo da obrišemo anotacije vezane za SpringBoot, test foldere, kao i application.properties.

```
KjarApplication.java X
kjar > src > main > java > com > ftn > sbnz > kjar > J KjarApplication.java > KjarApplication
1 package com.ftn.sbnz.kjar;
2
3 public class KjarApplication {
4
5     Run | Debug
6     public static void main(String[] args) {
7         System.out.println(x: "Hello from kjar!");
8     }
9 }
10
```

```
J KjarApplication.java J ModelApplication.java X
model > src > main > java > com > ftn > sbnz > model > J ModelApplication.java > Mo
1 package com.ftn.sbnz.model;
2
3 public class ModelApplication {
4
5     Run | Debug
6     public static void main(String[] args) {
7         System.out.println(x: "Hello from model!");
8     }
9 }
10
```

Kjar projekat

Zatim u *pom.xml* kjar projekta dodajemo zavisnosti kako bismo *Drools* mogli da koristimo. Unutar *properties* tagova možete da dodate `<drools.version>7.49.0.Final</drools.version>` koji ćete u ostatku pom-a koristiti.

```
<properties>
  <java.version>11</java.version>
  <drools.version>7.49.0.Final</drools.version>
</properties>
```

Zavisnosti koje se dodaju za Drools

```
<dependency>
  <groupId>org.kie</groupId>
  <artifactId>kie-ci</artifactId>
  <version>${drools.version}</version>
</dependency>

<dependency>
  <groupId>org.kie</groupId>
  <artifactId>kie-api</artifactId>
  <version>${drools.version}</version>
</dependency>

<dependency>
  <groupId>org.drools</groupId>
  <artifactId>drools-core</artifactId>
  <version>${drools.version}</version>
</dependency>

<dependency>
  <groupId>org.drools</groupId>
  <artifactId>drools-compiler</artifactId>
  <version>${drools.version}</version>
</dependency>
```

Povezivanje model projekta sa kjar projektom

Kako bi mogli koristiti klase iz **model** projekta, potrebno je da taj projekat kao zavisnost dodamo u pom.xml **kjar** projekta. Potrebno je da dodate **groupId**, **artifactId** i **version** model projekta u pom.xml kjar projekta.

```
<dependency>
  <groupId>com.ftn.sbnz</groupId>
  <artifactId>model</artifactId>
  <version>0.0.1-SNAPSHOT</version>
</dependency>
```

Figure 1 Pom.xml Kjar-a

Zatim je potrebno unutar resources foldera kreirati novi folder META-INF i unutar njega kmodule.xml fajl, u koji je potrebno upisati.

```
<kmodule xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://jboss.org/kie/6.0.0/kmodule">

</kmodule>
```



Model projekat

Ukoliko je potrebno da koristite anotacije vezane za *Drools*, potrebno je da dodate potrebne zavisnosti u pom.xml

Service projekat

Ovaj projekat je glavna aplikacija u kojoj koristimo klase iz *Model* projekta i pravila iz *Kjar* projekta. Dakle, potrebno je dodati zavisnosti ka tim projektima.

```
<dependency>
  <groupId>com.ftn.sbnz</groupId>
  <artifactId>kjar</artifactId>
  <version>0.0.1-SNAPSHOT</version>
</dependency>

<dependency>
  <groupId>com.ftn.sbnz</groupId>
  <artifactId>model</artifactId>
  <version>0.0.1-SNAPSHOT</version>
</dependency>
```

U main klasu u Service projektu, potrebno je dodati @Bean za KieContainer.

@Bean

```
public KieContainer kieContainer() {

    KieServices ks = KieServices.Factory.get();

    KieContainer kContainer = ks

        .newKieContainer(ks.newReleaseId("com.ftn.sbnz", "skjar", "0.0.1-SNAPSHOT"));

    KieScanner kScanner = ks.newKieScanner(kContainer);

    kScanner.start(1000);

    return kContainer;

}
```

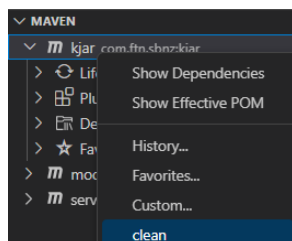
```
J ServiceApplication.java X
service > src > main > java > com > ftn > sbnz > service > J ServiceApplication.java > ...
1 package com.ftn.sbnz.service;
2
3 import org.kie.api.KieServices;
4 import org.kie.api.builder.KieScanner;
5 import org.kie.api.runtime.KieContainer;
6 import org.springframework.boot.SpringApplication;
7 import org.springframework.boot.autoconfigure.SpringBootApplication;
8 import org.springframework.context.annotation.Bean;
9
10 @SpringBootApplication
11 public class ServiceApplication {
12
13     Run | Debug
14     public static void main(String[] args) {
15         SpringApplication.run(primarySource: ServiceApplication.class, args);
16     }
17
18     @Bean
19     public KieContainer kieContainer() {
20         KieServices ks = KieServices.Factory.get();
21         KieContainer kContainer = ks
22             .newKieContainer(ks.newReleaseId(groupId: "com.ftn.sbnz", artifactId: "kjar", version: "0.0.1-SNAPSHOT"));
23         KieScanner kScanner = ks.newKieScanner(kContainer);
24         kScanner.start(pollingInterval: 1000);
25         return kContainer;
26     }
27 }
```

Pokretanje

Kako ne biste imali problema sa pozicioniranjem i pokretanjem, u folder u koji ste smestili projekte možete da prekopirate .mvn, mvnw i mvnw.cmd iz bilo kog projekta ([link](#))

.mvn	28.2.2023. 22:31	File folder	
.vscode	28.2.2023. 22:21	File folder	
kjar	28.2.2023. 21:53	File folder	
model	28.2.2023. 21:53	File folder	
service	28.2.2023. 21:53	File folder	
mvnw	28.2.2023. 20:51	File	11 KB
mvnw	28.2.2023. 20:51	Windows Comma...	7 KB

Prvo je potrebno da instalirate Kjar projekat. U okviru *EXPLORER* taba u *VS codu*, u donjem dijelu postoji kartica *MAVEN*, nakon što je otvorite biće prikazani projekti. Potrebno je desnim klikom na kjar da prvo izabere opciju **clean**.



Nakon što se clean izvrši, ponovo desnim klikom na kjar izaberete **install**.

Ukoliko bude grešku izbacivalo

.mvn\wrapper\maven-wrapper.properties.

Exception in thread "main" java.lang.RuntimeException: Wrapper properties file
'C:\Users\filip\Desktop\Nastava\SBNZ\Canvas\Vjezbe1\.mvn\wrapper\maven-wrapper.properties' does
not exist.

at org.apache.maven.wrapper.WrapperExecutor.forWrapperPropertiesFile(WrapperExecutor.java:65)

at org.apache.maven.wrapper.MavenWrapperMain.main(MavenWrapperMain.java:75)

Pokrenite komandu mvn -N io.takari:maven:wrapper u terminalu.