JAVA SPRING MICROSERVICES

6. forduló

A kategória támogatója: Vodafone Magyarország Zrt.

Ismertető a feladathoz

- Dr. Ellie Sattler: Istenem valami történt, valami baj van, valami nagy baj van.
- John Hammond: Csak apró késedelem, ez minden, minden nagy szórakoztató parkban előfordult. Amikor Disney Land megnyílt 1956-ban semmi sem működött.
- Dr. Ian Malcolm: Igen de ha defektet kap a szellemvasút nem falja fel a turistákat.

Mr. Hammond nem spórol semmin és a csapatnak több környezetet is finonszíroz a szolgáltatások futtatására, mindegyik combos vas, mint akár azok a dinoszauruszok, akiknek a felügyeletére készíti csapatotok az alkalmazást.

Mint tapasztalt fejlesztő tudod, hogy egy környezet nem biztosítja a fejlesztés során a mobilitást, az éles környezeten kívül a csapatotok elérte, hogy legyen egy teszt és egy úgynevezett "User acceptence test" környezet, amelyen ha Mr. Hammond ámént mond a lefejlesztett funkciókra, akkor azok kerülhetnek ki az éles környezetre.

Eddig ez mind szép és jó viszont, rendelkezünk már nagyon sok függőséggel és annál is több szenzitív adattal (adatbázis hozzáférések, külső API kulcsok satöbbi), ezek nem kerülhetnek csak úgy bele a végső alkalmazás kódba, plusz vajon hogyan is tudnánk különbséget tenni a környezetek között, ha minden szenzitív adat konstansként szerepelne a kódban?

Mire lesz szükséged a fordulóban?

- Internet kapcsolat
- Java 11 telepítve
- Maven telepítve (vagy Maven Wrapper használata)
- Fejlesztői környezet (ajánlott)
- Docker és/vagy docker-compose

Ha egy adott feladat egy annotációt vár válaszként, akkor oda csak az annotáció nevét add meg @ előjellel, nem szükséges teljes minősített nevet megadni, például: @NotNull.

Felhasznált idő: 30:00/30:00 **Elért pontszám:** 2/12

1. feladat 0/1 pont

Mi annak a változónak a neve (környezeti vagy rendszer) amely az értékét JSON formátumban várja és a Spring keretrendszer felolvassa és alkalmazza mint konfigurációs beállítás?

Válaszok

A helyes válasz:

SPRING_APPLICATION_JSON

spring_application_json

SPRING.APPLICATION.JSON

spring.application.json

SPRING_APPLICATION_JSON || spring.application.json

SPRING_APPLICATION_JSON environment prop, -Dspring.application.json system prop

Magyarázat

Az egyik helyes megoldás a SPRING_APPLICATION_JSON érték, viszont a Spring bármilyen megadott környezeti változót átalakít, nagybetűsre és lecseréli a benne lévő "." (pont) karaktereket "_" (aláhúzás) karakterekre és úgy olvassa fel a kontextusban, így a következők is helyesek:

- spring_application_json
- SPRING.APPLICATION.JSON
- spring.application.json

Hivatalos dokumentáció: https://docs.spring.io/spring-boot/docs/current/reference/html/features.html#features.external-config.application-json

2. feladat 0/1 pont

Az alkalmazásunk egy külső konfigurációs szervertől szerzi be a beállításait. Melyik az az annotáció, amivel az alkalmazásunk képes lesz egy adott bean osztályt érintő változásokat felismerni és alkalmazni futásidőben, ha az konfigurációs szerveren változás történik?

Válaszok

A helyes válasz:

@RefreshScope

Magyarázat

Hivatalos dokumentációt: https://docs.spring.io/spring-cloud-commons/docs/3.1.4/reference/html/#refresh-scope

3. feladat 2/2 pont

Spring Boot-on belül a több forrásból dolgozó konfigurációs paraméterek feloldásának van egy prioritási sorrendje, ahol a magasabb prioritással rendelkező forrás felülírhatja az azonos kulccsal rendelkező alacsonyabb prioritású forrás értékét.

Melyik a helyes erősségi sorrend? (első elem a legerősebb, az felül tudja írni az utána lévő forrásban tárolt kulcs értékét)

1. Szervlet konfigurációs paraméter
2. Java Rendszer Beállítások
3. JNDI paraméterek
4. Konfigurációs adatok (application[-{profile}]+.[yml|properties])
5. @PropertySource annotációban feloldott konfigurációs állomány
6. Rendszer szintű környezeti változók

Válasz

- [ ] 3,1,2,6,4,5
- [x] 1,3,2,6,4,5
  Ez a válasz helyes, és meg is jelölted.
- [ ] 6,2,3,1,5,4

Magyarázat

Hivatalos dokumentáció: https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/#features.external-config

4. feladat 0/2 pont

Melyik környezeti változóval állíthatjuk be az futó alkalmazás profilját a **prodredis** és **proddb** értékekre?

Válasz

- [ ] **SPRING_PROFILE_ACTIVE[0]=prodredis;SPRING_PROFILE_ACTIVE[1]=proddb**
- [ ] **SPRING_ACTIVE_PROFILE=prodredis,proddb**
- [x] **SPRING_ACTIVE_PROFILES=prodredis,proddb**
  Ez a válasz helytelen, de megjelölted.
- [x] **SPRING_PROFILES_ACTIVE=prodredis,proddb**
  Ez a válasz helyes, de nem jelölted meg.

Magyarázat

Hivatalos dokumentáció: https://docs.spring.io/spring-boot/docs/current/reference/htmlsingle/#howto.properties-and-configuration.set-active-spring-profiles

5. feladat 0/2 pont

Dennis Nedry az előző alkalmazott egy konfigurációs szerverre másolta fel a területeket kezelő szolgáltatás konfigurációs beállításait.

Külső alkalmazás elérése:

- A konfigurációs szervert a következő URL-en lehet elérni: **https://config.ingensol.co**
- Indítsd el lokálisan a szolgáltatást a következő Docker képfájllal: ghcr.io/ingen-sol/ingen-sol-config-server - a 8080-as porton keresztül érhető el. - docker run -p 8080:8080 ghcr.io/ingen-sol/ingen-sol-config-server

Amit tudunk az az, hogy a konfigurációs szerver egy úgynevezett "Basic authentication" módszerrel van védve, amelyet Dennis asztalán a nagy kuplerájban egy cetlin találtatok meg.

- Felhasználónév: dennis-nedry
- Jelszó: i-truly-love-money

Készíts egy alkalmazást amely a spring-cloud-starter-config függőséget használja és teljesítsd a következő feladatokat! (Gyors projekt összeállítás)

Készíts egy alkalmazást amely a **Spring Cloud Config** függőséget használja, kösd be a fentebb említett konfigurációs szervert és válszolj a következő kérdésekre, a **production** konfiguráció a következő alkalmazásra érhető el: **territory-service**.

Az 5.-7. kérdéseket a fentieket használva tudod megoldani!

A konfigurációs állományon belül van egy titkos kulcs, amely fel van bontva több részre: mi a konfiguráció neve? Kérlek válaszodat az index szám és az utolsó kötőjel nélkül add meg!

Válasz

A helyes válasz:

secret-key-chunk

Magyarázat

A feladatból kiderül, hogy a konfigurációs szerveren léteznie kell egy fájlnak, aminek az a neve, hogy **territory-service-production.yml.** Miután ezt a két dolgot összeraktuk, akkor a konfigurációs szerveren a következő URL-re kell elmennünk, hogy szemügyre vegyük a konfigurációs állományt: https://config.ingensol.co/territory-service/production (vagy a versenyző gépén futatott konténer segítségével, ha a megadott parancsal indította el: http://localhost:8080/territory-service/production) - Itt kiderül, hogy a kulcs amit keresünk az a **secret-key-chunk**

6. feladat 0/1 pont

Hány részre lett darabolva a kulcs?

Válasz

A helyes válasz:

49

Magyarázat

Miután megvan a kulcs neve, meg kell néznünk, hogy hány részre lett feldarabolva. Ez az érték 49.

7. feladat 0/3 pont

Hogyha visszafejted a paraméterekben megadott kulcsok értékét és egybefűzöd, akkor annak mi lesz az értéke?

Válasz

A helyes válasz:

dennis-nedry-and-ray-arnold-were-not-good-friends

Magyarázat

```
spring.application.name=territory-service
spring.profiles.active=production
spring.config.import=optional:configserver:https://dennis-nedry:i-truly-love-money@config.ingensol.co

# Vagy ha a versenyző gépén volt indítva a konfigurációs alkalmazás: optional:configserver:http://dennis-nedr

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.ApplicationArguments;
import org.springframework.boot.ApplicationRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.core.env.Environment;

import java.util.Base64;

@SpringBootApplication
public class ConfigClientApplication implements ApplicationRunner {

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

    public static final int NUMBER_OF_CHUNKS = 48;
    public static final String KEY_PREFIX = "secret-key-chunk-";

    @Autowired
    private Environment environment;

    @Override
    public void run(ApplicationArguments args) {
        StringBuilder result = new StringBuilder();
        for (int i = 0; i <= NUMBER_OF_CHUNKS; i++) {
            String key = KEY_PREFIX + i;
            String value = environment.getProperty(key);
            result.append(new String(Base64.getDecoder().decode(value)));
        }
        System.out.println(result);
    }

}
```

Megjelenés

Világos