JAVA SPRING MICROSERVICES

7. forduló

A kategória támogatója: Vodafone Magyarország Zrt.

Ismertető a feladathoz

- Dr. Ellie Sattler: Nekem nem tiszta ez a káosz.
- Dr. Ian Malcolm: Egész egyszerűen a komplex rendszerek kiszámíthatatlanságáról van szó.

Dr. Hammond alig várja az éles indulást, viszont még mindig van tennivaló, a konfigok már ki vannak telepítve, viszont vannak bizonyos külső szolgáltatások amelyek egyelőre nem elég stabilak és veszélyeztetik a park rendszereit.

A szolgáltatásainkba szeretnénk több olyan függőséget integrálni, amelyek a szolgáltatások közötti megfigyelhetőséget és átláthatóságot támogatják, emellett a Spring Boot Actuatort is. A következő feladatokban ezen függőségekről lesz szó.

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: 00:00/30:00 **Elért pontszám:** 0/13

1. feladat 0/1 pont

A következő függőségek közül melyik biztosít számunkra egy API-t, amivel megvalósíthatunk **elosztott követést** a Springes ökoszisztémában?

Válasz

- ( ) Spring Cloud Bus
- ( ) Spring Cloud Contract
- (•) Spring Cloud Sleuth
  Ez a válasz helyes, de nem jelölted meg.
- ( ) Spring Cloud Zookeeper

Magyarázat

A felsoroltak közül a Spring Cloud Sleuth a megoldás, a többi úgyszint létező függőség, de a kérdéses API-t a **Spring Cloud Sleuth** biztosítja.

Hivatalos dokumentáció: https://docs.spring.io/spring-cloud-sleuth/docs/current/reference/html/getting-started.html#getting-started-introducing-spring-cloud-sleuth

2. feladat 0/1 pont

A szolgáltatásainkban szeretnénk minél több információt gyűjteni a folyamatokról, és emiatt integrálásra került a **Micrometer** függőség (Tekintsünk el a mögöttes rétegtől, hogy mi is lesz, ami gyűjti a metrikákat).

Szeretnénk kézzel készített metrikákat gyűjteni az egyik szolgáltatás osztályban, amikor annak az egyik metódusa meghívásra kerül. **Melyik interfészen/osztályon/absztrakt osztályon keresztül tudjuk ezt megtenni?** (Csak a nevére kell fókuszálni,nem kell teljes minősített nevet megadni.)

Válaszok

A helyes válasz:

MeterRegistry

io.micrometer.core.instrument.MeterRegistry

Meter

io.micrometer.core.instrument

Counter.increment(); meterRegistry.counter

Magyarázat

Amint integráltuk a **Micrometer** függőséget, a fejlesztők képesek lesznek a ***MeterRegistry*** absztrakt osztály használatára, amely egynél több implementációval rendelkezik, de a Spring Boot adta auto-konfigurációs képességeinek megfelelően a szolgáltatás indulásakor a háttérben a megfelelő osztályokat példányosítja, és nekünk "végfelhasználóknak" az előbb említett absztrakt osztályon keresztül ad lehetőséget metrikák kézi vezérlésére.

Hivatalos dokumentáció: https://docs.spring.io/spring-boot/docs/current/reference/html/actuator.html#actuator.metrics.getting-started

3. feladat 0/1 pont

Mi annak az aktuátor végpontnak a neve, amely betekintést enged az alkalmazás által használt környezeti változókra?

Válaszok

A helyes válasz:

env

/actuator/env

/env

Magyarázat

Az **env** nevű végpont engedélyezésével az alkalmazásunk által használt konfigurációs forrásokat és a különböző konfigurációs kulcsok értékeit tekinthetjük meg.

Hivatalos dokumentáció: https://docs.spring.io/spring-boot/docs/current/reference/html/actuator.html#actuator.endpoints

4. feladat 0/1 pont

Hogyan tudjuk az előző feladatban lévő aktuátor végpontot engedélyezni, hogy a megfelelő végponton elérjük? (Kérlek a válaszodat úgy add meg mintha egy.properties fájlban írnád. Például: ez.itt.a.kulcs=ez-pedig-az-érték)

Válaszok

A helyes válasz:

management.endpoints.web.exposure.include=env

MANAGEMENT.ENDPOINTS.WEB.EXPOSURE.INCLUDE=env

management.endpoint.env.enabled=true

management.endpoint.env.enabled=true vagy management.endpoints.web.exposure.include=env

Magyarázat

Az aktutátor függőségen belül alapértelmezetten 1 végpont van nyitva a webes csatornán keresztül, ez pedig a health. JMX-es csatornán keresztül mindegyik nyitva van amelyik támogatott. Ahhoz, hogy a feladatban kért aktuátort engedélyezni tudjuk, ahhoz tudnunk kell, hogy milyen kulcs értékkel tudjuk manipulálni az engedélyezett aktuátorokat, és tudnunk kell a nevét is. A ***management.endpoints.web.exposure.include***, vagy nagybetűsen ***MANAGEMENT.ENDPOINTS.WEB.EXPOSURE.INCLUDE*** kulcs módosításával érhetjük el a kért változást, az értékének pedig az ***env***-et kell adni. Ha az ENV-et nagybetűvel adjuk meg, akkor a Spring induláskor egy hibával fogja ezt viszonozni.

Hivatalos dokumentáció: https://docs.spring.io/spring-boot/docs/current/reference/html/actuator.html#actuator.endpoints.exposing

5. feladat 0/2 pont

Mely aktuátor végpontok érvénytelenek (nem léteznek)?

Válaszok

- [x] kubernetes
  Ez a válasz helyes, de nem jelölted meg.
- [ ] info
- [x] mvc

Ez a válasz helyes, de nem jelölted meg.

Magyarázat

Kedves Versenyzők!

A feladat írásakor a Spring Boot 2-es verziója számított a legutolsó friss verziónak, de a Spring Boot 3 megjelenésével, amely november 24-én történt meg, a jolokia kikerült az Actuator végpontok listájából, így ezt a válaszlehetőséget töröltük.

Köszönjü megértéseteket!

A megadott listában a **kubernetes** mint aktuátor végpont nem létezik, viszont a Spring Boot Actuator függőség rendelkezik egy a Kubernetes számára hasznos végpont kombinációval.

Az **mvc** végpont sem létezik, a másik kettő, az **info** és a **jolokia** igen.

Hivatalos dokumentáció: https://docs.spring.io/spring-boot/docs/current/reference/html/actuator.html#actuator.endpoints

6. feladat 0/1 pont

Egy belső InGen Sol-os szolgáltatás a belső helyiségek biztonságáért felel, a parkon belül sok ilyen található. A fejlesztők elmondása alapján a Spring Boot Actuatort használták a helyiségek menedzselésére.

Külső alkalmazás elérése:

- Az alkalmazást a következő URL-en lehet elérni: **https://internal-management.ingensol.co**
- Indítsd el lokálisan a szolgáltatást a következő Docker képfájllal: ghcr.io/ingen-sol/ingen-sol-internal-management-service - a 8080-as porton keresztül érhető el. **- docker run -p 8080:8080** ghcr.io/ingen-sol/ingen-sol-internal-management-service

A 6.-10. kérdések megválaszolásához a fentieket használd!

Milyen aktuátor végponton érhető el a másik szolgáltatást fejlesztő csapat helyiségeket listázó funkciója? (Ha a végpontot a http://szerver:port/elso/masodik néven érhető el, akkor a válaszként az **/elso/masodik** értéket írd.)

Válasz

A helyes válasz:

/actuator/rooms

Magyarázat

Ha megnyitjuk az adott linket, vagy saját kezűleg indítjuk el a Docker konténert, akkor utána az **/actuator** végponton listázódik az összes elérhető aktuátor, jelenleg csak a **rooms** nevű aktuátor érhető el, az pedig az **/actuator/rooms** végponton.

7. feladat 0/2 pont

Dennis Nedry helyisége egy igazán különleges hely, mi miatt?

Válasz

- Mr. Hammond bankkártyáinak az adatai megvannak neki egy cetlin az asztalán a kuplerájban
- Ray Arnold-ot ábrázoló poszter van a falán, amelybe több darts nyíl van már beledobva

Ez a válasz helyes, de nem jelölted meg.

- A következő park generációjának egy módosított kódja található meg egy Floppy lemezen
- Egy raptor tojást rejtett a szekrényébe

Magyarázat

Miután megnyitottuk az **/actuator/rooms** végpontot, akkor a különböző helyiségek listázódnak ki elénk, a feladat arról szól, hogy miért különleges Dennis Nedry szobája. Ezt csak úgy tudhatjuk meg ha felismerjük, hogy van egy másik végpont, ami egy **{identifier}** nevű paramétert kaphat meg. Ha ide Dennis szobájának az azonosítóját írjuk be, akkor fixen látni fogjuk, hogy miért is különleges. Ezt csak úgy tudjuk megtenni, ha ismerjük, hogy egy saját **aktuátor** végpont hogyan működik, a hivatalos dokumentáció szerint **GET**, **POST** és **DELETE** HTTP metódusokkal tudunk interakcióba lépni ezekkel a végpontokkal: https://docs.spring.io/spring-boot/docs/current/reference/html/actuator.html#actuator.endpoints.implementing-custom.

A szobájában egy Ray Arnold poszter van a falon, ez az információ pedig az **actuator/rooms/DENNIS-NEDRY-PLAYGROUND** oldalon található **details** nevű mezőből olvasható ki.


```
{
   "room": {
      "identifier": "DENNIS-NEDRY-PLAYGROUND",
      "shortName": "Dennis Nedry Playground",
      "longName": "Dennis Nedry's playground, where he works on his side projects after worktime",
      "doorState": "CLOSED"
   },
   "details": [
      {
         "property": "RAY-ARNOLD-POSTER-WITH-A-DART-ARROW",
         "place": "WALL"
      }
   ]
}
```


8. feladat 0/1 pont

Ahhoz, hogy ezeket az információkat megtudjuk, milyen végpontot kell meghívnunk? (Kérlek a válaszodat az előző kérdésben említett módon add meg!)

Válasz

A helyes válasz:

/actuator/rooms/DENNIS-NEDRY-PLAYGROUND

Magyarázat

A hívandó végpont: **/actuator/rooms/DENNIS-NEDRY-PLAYGROUND** - ez egy **GET**-es hívás.

9. feladat 0/1 pont

A helyiségek ajtajai nyithatóak és zárhatóak, viszont van egy, amely az utóbbi tesztek folyamán mindig bekrepált. Melyik ez az ajtó? (Kérlek válaszodban az identifier mező értékét írd!)

Válasz

A helyes válasz:

HIDDEN-LAB-01

Magyarázat

A művelet a leírás alapján valaminek az állapotát változtatná meg, mégpedig az ajtó állapotát, ennek a mezőnek a neve a **doorState. A**hhoz, hogy minden ajtót végig próbáljunk, egy **POST**-os hívást kell indítanunk, ahol vagy a kérés törzsében egy JSON-ben vagy query paraméterként megadjuk a **doorState** nevű mezőt, aminek egy másik értéket állítottunk be, vagy **CLOSED**, vagy **OPEN**. Ennek hatására az ajtók állapotát tudjuk frissíteni, viszont a **HIDDEN-LAB-01** azonosítójú helység ajtaját nem tudjuk felülírni.

```
POST /actuator/rooms/HIDDEN-LAB-01?doorState=CLOSED
```

10. feladat 0/2 pont

Mi az állapota ennek az ajtónak, ha megpróbáljuk megváltoztatni? (A válaszban a **doorState** nevű mező értékét keressük!)

Válasz

A helyes válasz:

OVERLOADED

Magyarázat

A **POST**-os hívás után fogjuk látni, hogy a **doorState** nevű mező értéke OVERLOADED, ez egy titkos helyiség, nem tudunk bejutni.

Egy lehetséges Java alapú megoldás Feign kliens használatával:

```
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.cloud.openfeign.EnableFeignClients;
import org.springframework.cloud.openfeign.FeignClient;
import org.springframework.http.MediaType;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestParam;

import java.util.List;
```

```java
@SpringBootApplication
@EnableFeignClients
public class RoundSolution implements ApplicationRunner {


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

    @Autowired
    private ExternalClient externalClient;

    @Override
    public void run(ApplicationArguments args) throws Exception {
        List<String> rooms = List.of("RAPTOR-HOUSE-01", "LAB-01", "RAY-ARNOLD-CHILL-BAY", "LAB-02", "WAREHOUS
        for (String room : rooms) {
            try {
                Object roomResponse = externalClient.getRoom(room);
                System.out.println(roomResponse);
                for (DoorState doorState : DoorState.values()) {
                    Object postRoomResponse = externalClient.postRoom(room, doorState.name());
                    System.out.println(postRoomResponse);
                }
                System.out.println("----");
            } catch (Exception e) {
                System.err.println("Hiba a hívás során: " + e);
            }
        }
    }
}

enum DoorState {
    OPEN,
    CLOSED
}

@FeignClient(name = "internal-management-supply", url = "https://internal-management.ingensol.co")
interface ExternalClient {

    @GetMapping(consumes = {MediaType.APPLICATION_JSON_VALUE}, path = "/actuator/rooms/{identifier}")
    Object getRoom(@PathVariable("identifier") String identifier);

    @PostMapping(consumes = {MediaType.APPLICATION_JSON_VALUE}, path = "/actuator/rooms/{identifier}")
    Object postRoom(@PathVariable("identifier") String identifier, @RequestParam("doorState") String roomStat

}
```

Kimenet:

```
{room={identifier=RAPTOR-HOUSE-01, shortName=Raptor house 01, longName=Raptor house on Isla Nublar, must be a
{room={identifier=RAPTOR-HOUSE-01, shortName=Raptor house 01, longName=Raptor house on Isla Nublar, must be a
{room={identifier=RAPTOR-HOUSE-01, shortName=Raptor house 01, longName=Raptor house on Isla Nublar, must be a
----
{room={identifier=LAB-01, shortName=Laboratory, longName=Dinosaur research laboratory, doorState=OPEN}, detai
{room={identifier=LAB-01, shortName=Laboratory, longName=Dinosaur research laboratory, doorState=OPEN}, messa
{room={identifier=LAB-01, shortName=Laboratory, longName=Dinosaur research laboratory, doorState=CLOSED}, mes
----
{room={identifier=RAY-ARNOLD-CHILL-BAY, shortName=Ray Arnold Bay, longName=Ray Arnold's chill bay where he re
{room={identifier=RAY-ARNOLD-CHILL-BAY, shortName=Ray Arnold Bay, longName=Ray Arnold's chill bay where he re
{room={identifier=RAY-ARNOLD-CHILL-BAY, shortName=Ray Arnold Bay, longName=Ray Arnold's chill bay where he re
----
{room={identifier=LAB-02, shortName=Small Laboratory, longName=Small research laboratory, doorState=OPEN}, de
{room={identifier=LAB-02, shortName=Small Laboratory, longName=Small research laboratory, doorState=OPEN}, me
```

```
{room={identifier=LAB-02, shortName=Small Laboratory, longName=Small research laboratory, doorState=CLOSED},
----
{room={identifier=WAREHOUSE-07, shortName=Warehouse on Isla Nublar, longName=Warehouse with full of weapons a
{room={identifier=WAREHOUSE-07, shortName=Warehouse on Isla Nublar, longName=Warehouse with full of weapons a
{room={identifier=WAREHOUSE-07, shortName=Warehouse on Isla Nublar, longName=Warehouse with full of weapons a
----
{room={identifier=HIDDEN-LAB-01, shortName=Hidden Lab 01, longName=Hidden Laboratory in the F01 building with
{room={identifier=HIDDEN-LAB-01, shortName=Hidden Lab 01, longName=Hidden Laboratory in the F01 building with
{room={identifier=HIDDEN-LAB-01, shortName=Hidden Lab 01, longName=Hidden Laboratory in the F01 building with
----
{room={identifier=HAMMOND-OFFICE, shortName=Mr. Hammond's Lab, longName=Mr. Hammond's CEO Office room where h
{room={identifier=HAMMOND-OFFICE, shortName=Mr. Hammond's Lab, longName=Mr. Hammond's CEO Office room where h
{room={identifier=HAMMOND-OFFICE, shortName=Mr. Hammond's Lab, longName=Mr. Hammond's CEO Office room where h
----
{room={identifier=DENNIS-NEDRY-PLAYGROUND, shortName=Dennis Nedry Playground, longName=Dennis Nedry's playgro
{room={identifier=DENNIS-NEDRY-PLAYGROUND, shortName=Dennis Nedry Playground, longName=Dennis Nedry's playgro
{room={identifier=DENNIS-NEDRY-PLAYGROUND, shortName=Dennis Nedry Playground, longName=Dennis Nedry's playgro
----
```

Lényeges kimenetek:

Dennis szobája:

```
{room={identifier=DENNIS-NEDRY-PLAYGROUND, shortName=Dennis Nedry Playground, longName=Dennis Nedry's playgro
```

HIDDEN-LAB-01-es szoba ajtó státuszának változtatására vonatkozó bejegyzés:

```
{room={identifier=HIDDEN-LAB-01, shortName=Hidden Lab 01, longName=Hidden Laboratory in the F01 building with
```