Začíname s Apache Kafka

Apache Kafka je mnoho vecí:

- distribuovaná platforma pre streaming dát
- message broker ako prostredník pre výmenu správ medzi komponentami distribuovaného systému

Podrobnosti nájdeme v slajdoch.

Ingrediencie

- Docker — pre spustenie Kafky
- Conduktor — grafický nástroj pre Kafku
- Spring Boot — pre rýchle vytvorenie producentov a konzumentov
- IntelliJ IDEA — IDE pre kód v Jave

Spúšťame Kafku

Kafka pozostáva z dvoch vrstiev:

- Apache Zookeper — podvozok slúžiaci pre chod a konfiguráciu distribuovaných služieb. Nebudeme ho bližšie rozoberať, pretože je o úroveň nižšie, ako potrebujeme pre tento tutoriál.
- samotná Kafka bežiaca nad Zookeeperom

Kafku je najjednoduchšie spustiť pomocou nástroja **Docker**, resp. **Docker Compose**, čím si ušetríme množstvo konfigurácie.

Docker Compose

1. Stiahnime si infraštruktúrny súbor `docker-compose.yml`.
2. Spustime oba dockerovské kontajnery:

   ```
   docker-compose up
   ```

3. Spustí sa Kafka v Dockeri a v logu uvidíme úspech:

   ```
   kafka_1      | [2020-11-14 13:44:44,041] INFO [KafkaServer id=1] started
   (kafka.server.KafkaServer)
   ```

Práca s Conduktorom

Stiahneme si Conduktor, ktorý v bezplatnej verzii podporuje cluster Kafky s jedným brokerom, čo postačí pre účely tutoriálu.

Vytvoríme nový cluster Kafky (**New Kafka Cluster**) s nasledovnými parametrami

Cluster Name

ľubovoľné meno, napríklad `local-docker`

BootStrap Servers

nastavme na `localhost:29092` ako jediný uzol, z ktorého sa štartuje cluster. Pozor! Naša Kafka v Dockeri používa neštandardný port.

Zookeeper

nastavme na `localhost:2181`

Konektivitu môžeme otestovať pomocou tlačidla **Test ZK**, resp **Test Kafka Connectivity**.

Producenti a konzumenti

Producentov a konzumentov vytvoríme pomocou Spring Bootu.

V IntelliJ IDEA vytvorme nový prázdny projekt (**File** | **New** | **Empty project**), do ktorého dodáme dva moduly (producentský a konzumentský).

Producentský modul

Vytvorenie modulu

Vytvorme nový modul (**File** | **New** | **Module**) typu **Spring Initializr**. Vyplňme názov skupiny (*Group*) a artefakt nazvime napríklad `producer`. V dialógovom okne následne vyberme z palety Spring Bootu súčasť **Spring for Apache Kafka** (sekcia **Messaging**).

NOTE	Alternatívne využime projekt https://start.spring.io/, kde si vieme vybrať súčasti, vygenerovať projekt v balíčku ZIP, stiahnuť a následne importovať.

Najdôležitejšou závislosťou je `spring-kafka`:

```
<dependency>
    <groupId>org.springframework.kafka</groupId>
    <artifactId>spring-kafka</artifactId>
</dependency>
```

Konfigurácia modulu

Keďže náš broker Kafky beží na neštandardnom porte, musíme to nakonfigurovať v producentovi. V súbore `src/main/resources` existuje konfiguračný súbor pre Spring Boot `application.properties`, do ktorého uveďme:

```
spring.kafka.producer.bootstrap-servers=localhost:29092
```

Kód modulu

Na komunikáciu s Kafkou sa využíva `KafkaTemplate`, ktorú si môžeme nechať autowirenúť / injectnúť do triedy aplikácie.

```
@Autowired
KafkaTemplate<String, String> kafka;
```

Trieda má dva generické parametre: pre kľúče (*keys*) a hodnoty (*values*). V tomto prípade budeme posielať reťazce.

Vďaka Spring Bootu je mnoho vecí už nakonfigurovaných a sústredíme sa len na to podstatné — odosielanie správ.

Ukážka metódy, ktorá periodicky odosiela správy:

```
public void publish() {
    kafka.send("payment", "Paid some money " + Instant.now());
}
```

Metóda `send` potrebuje dva parametre: názov *topicu*, do ktorého sa pošle správa a samotná správa. Keďže Kafka považuje správy za polia bajtov, treba sa postarať o prevod, čo zabezpečí Spring Boot a jeho integrácia s Kafkou.

Spustenie producenta

Keďže aplikácia v Spring Boote má štandardnú metódu `main`, spustíme ju triviálne. Pripojí sa k brokeru a vypublikuje doň správu do topicu `payment`.

IMPORTANT	Broker je štandardne pripravený automaticky vytvárať topicy pri prvom publikovaní správy.

NOTE	V ukážkovom kóde na GitHube uvidíme periodické zasielanie správy.

Kontrola v Conduktore

V Conduktore môžeme zistiť, že vznikol topic `payment` a produkujú sa doňho správy: počet (*Count*) postupne narastá.

Konzumentský modul

Príprava a konfigurácia

Podobne ako pri konzumentovi vytvorme nový modul (**File** | **New** | **Module**) typu **Spring Initializr**. Vyplňme názov skupiny (*Group*) a artefakt nazvime napríklad `producer`. V dialógovom okne následne vyberme z palety Spring Bootu súčasť **Spring for Apache Kafka** (sekcia **Messaging**).

Rovnako nakonfigurujme cestu k bootstrapovému serveru v `application.properties` identickým spôsobom: do `application.properties` dajme

```
spring.kafka.consumer.bootstrap-servers=localhost:29092
```

Kód konzumenta

Kód konzumenta využíva anotáciu `@KafkaListener`:

```
@KafkaListener(topics = "payment", ①
               groupId = "bank-branch") ②
public void consume(String message) {
    logger.info("Received: {}", message);
}
```

① Uvedieme topic, z ktorého budeme konzumovať.

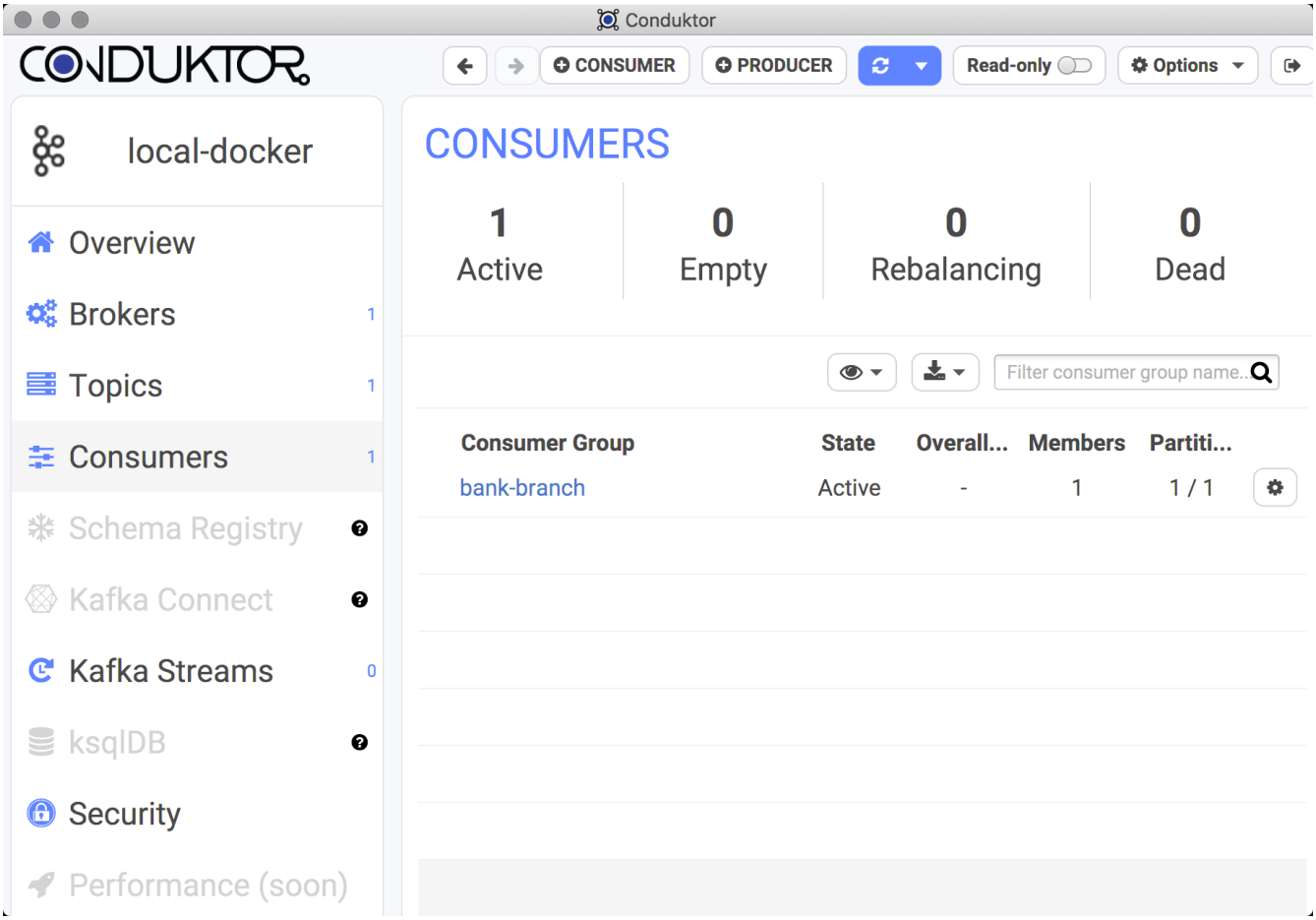
② Uvedieme identifikátor pre skupinu konzumentov (*consumer group*), do ktorej bude konzument patriť.

Spúšťame konzumenta

Konzumenta spustíme obvyklým spôsobom, cez spustenie hlavnej metódy. Sledujme, ako prijíma údaje z topicu.

Konzumenti v Conduktore

V Conduktore uvidíme, že v sekcii *Consumers* pribudla skupina (*Consumer Group*) s názvom `bank-branch`,ktorá má jediného člena.



Viacerí konzumenti a rebalance

Vytvorme teraz v IntelliJ IDEA ešte jednu konfiguráciu pre druhého konzumenta. Zvoľme z menu **Run | Edit Configurations**, vyberme z ľavého zoznamu v sekcii **Spring Boot** existujúcu konfiguráciu pre konzumenta a duplikujme ju. Nazvime ju odlišným spôsobom — napr. `Consumer #2`.

Spusťme teraz oboch konzumentov, najprv jedného a potom druhého.

Pri druhom konzumentovi uvidíme v jeho logu hlášku indikujúcu partíciu, na ktorú sa pripojil.

Log druhého konzumenta

```
2020-11-14 17:16:40.515  INFO 11741 --- [ntainer#0-0-C-1]
o.s.k.l.KafkaMessageListenerContainer    : bank-branch: partitions assigned: [payment-
0]
```

Zároveň uvidíme, že prvému konzumentovi bola odobratá partícia:

Log prvého konzumenta

```
2020-11-14 17:16:40.496  INFO 11739 --- [ntainer#0-0-C-1]
o.s.k.l.KafkaMessageListenerContainer    : bank-branch: partitions assigned: []
```

WARNING	Automatické vytváranie topicov cez Spring Boot pridelí takémuto topicu len jednu partíciu, čo znamená, že z nej môže konzumovať maximálne jeden člen skupiny konzumentov. Inak povedané, nevieme zabezpečiť distribuovanie práce.

Zvýšenie počtu partícií

V Kafke je možné dodatočne navyšovať počet partícií v topicu (nie však znižovať!). V Conduktore kliknime na topic `payment`, a cez tlačidlo **Advanced** a možnosť **Add Partition** dodajme rovno dve nové partície, t. j. nastavme tri partície.

Konzumentov nemusíme zastavovať, pretože rovno uvidíme, že nastal **rebalance** a podľa okolností jeden konzument vyfasoval dve partície a druhý konzument jednu partíciu.

TIP	Pohrajte sa so zastavovaním konzumentov a sledujte, ako konzumujú správy. Za štandardných okolností bude pri dvoch konzumentoch rovnomerné rozdelenie správ, približne na striedačku.