Spring Cloud Bus

Spring Cloud Bus

- Dağıtık mimarideki uygulamalar için lightweight bir **message broker kabiliyeti** sunar
- Uygulamalar **birbirleri ile haberleşmek** için bu message broker'ı kullanabilirler
- Bu broker aracılığı ile uygulamalar, meydana gelen **state değişiklikleri**nden diğer uygulamaları haberdar edebilirler
- Transport katmanı olarak **AMQP** veya **Kafka** kullanılabilir

Spring Cloud Bus

pom.xml

```
</dependencies>
    <dependency>
        <groupId>org.springframework.cloud</groupId>
        <artifactId>spring-cloud-starter-bus-amqp</artifactId>
    </dependency>
</dependencies>
```

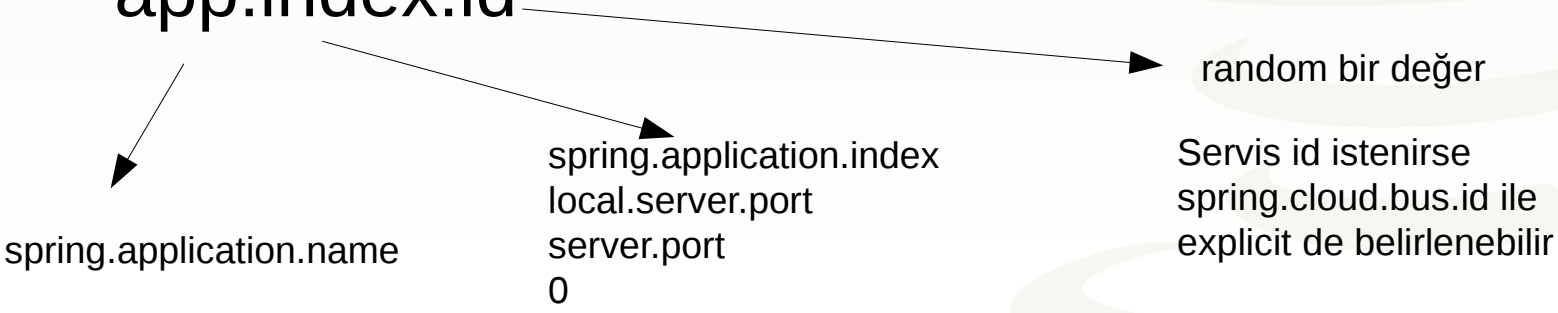
application.properties

```
spring.application.name=fooservice

spring.rabbitmq.host=localhost
spring.rabbitmq.port=5672
spring.rabbitmq.username=guest
spring.rabbitmq.password=guest
```

Spring Cloud Bus

- Event'ler, bus'ı dinleyen **bütün** node'lara yada **spesifik bir node**'a gönderilebilir
- Ya da **belirli bir servis**'e de gönderilebilir
- Burada **servis adı** Eureka ile belirlenen isimdir
- Her bir node bir **servis ID** ile tespit edilir
  - app:index:id



Spring Cloud Bus Endpoints

- Spring Cloud Bus **iki endpoint** sağlar
  - /actuator/bus-refresh
    - **@RefreshScope** cache'leri reset'ler ve **@ConfigurationProperties** bean'larını rebind eder
  - /actuator/bus-env
    - **POST request** ile belirli bir **property değeri**ni bütün node'larda günceller

```
{
        "name": "key1",
        "value": "value1"
}
```

    - /bus-env/fooservice:** ile güncelleme fooservice instance'larına spesifik yapılabilir

Spring Cloud Bus Endpoints

pom.xml

```
</dependencies>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-actuator</artifactId>
    </dependency>
</dependencies>
```

application.properties

```
management.endpoints.web.exposure.include=bus-refresh,bus-env
```

RemoteApplicationEvent

- Spring Bus event'leri **RemoteApplicationEvent** sınıfından türerler
- Event instance'ları default olarak **JSON formatı**na dönüştürülerek taşınır
- Event sınıfları ilgili **bütün uygulamalar tarafından paylaşılmalı**dır

RemoteApplicationEvent

- **@RemoteApplicationEventScan** anotasyonu ile uygulama içerisinde remote event sınıflarının hangi paketler altında yer aldığı belirtilmelidir

```
@RemoteApplicationEventScan({"com.javaegitimleri"})
@SpringBootApplication
public class HelloServiceApplication {
    ...
}
```

RemoteApplicationEvent Örneği

```
public class TestEvent extends RemoteApplicationEvent {
     private String message;

     public TestEvent(Object source, String originService,
                                          String destinationService) {
          super(source, originService, destinationService);
}

     public TestEvent(Object source, String originService) {
          super(source, originService);
     }

     public String getMessage() {
          return message;
     }

     public void setMessage(String message) {
          this.message = message;
     }
}
```

Event Publisher

```
@Component
public class TestEventPublisher {

     @Autowired
     private ApplicationContext applicationContext;

     @Autowired
     private ServiceMatcher serviceMatcher;


     public void publishEvent() {
          TestEvent event = new TestEvent(this,serviceMatcher.getServiceId());
          event.setMessage("Message created at :" + new Date());

          applicationContext.publishEvent(event);
     }
}
```

Event Listener

```
@Component
public class TestEventListener {

     @EventListener
     public void handle(TestEvent event) {
          System.out.println(">>>Received event with message :"
                                                  + event.getMessage());
     }

}
```

İletişim

- Harezmi Bilişim Çözümleri
- Kurumsal Java Eğitimleri
- http://www.java-egitimleri.com
- info@java-egitimleri.com