TP 08 - Identifiant de corrélation et ELK

8.0 - Objectifs

Estimated Time : **60** *minutes*

Le présent TP se veut présenter comment mettre en place la suite ELK et comment implémenter un identifiant de corrélation technique.

- Utilisation MDC SLF4J
- Implémentation d'un identifiant de corrélation
- Afficher l'identifiant de corrélation dans les journaux
- Mise en place d'Elasticsearch et Logstash via docker
- Exploration des données via Kibana
- Création d'un grok personnalisé

8.1 - Mise en place d'un identifiant de corrélation

- Dans chaque projet contenant des controlleurs ajouter un filtre de requêtes HTTP s'inspirant du cours.
- Nommer la classe : `package com.zenika.microservices.resanet.filter.CorrelationHeaderFilter`
- Notes :
  - La partie async du cours n'est pas nécessaire.

8.2 - Logback configuration

- Ajouter après la date l'identifiant de corrélation comme indiqué dans le cours dans le fichier de configuration **logback.xml**

8.3 - Tester

- Tester visuellement que cela marche :
  - `mvn clean install && docker-compose up`

    - `curl http://127.0.0.1:8081/ville`

- Le resultat attendue est le suivant :

```
resanet-catalogue_1    | 2016-02-03 14:51:49  INFO  c.z.m.r.c.s.Applicat
ion:48 - Starting Application v0.0-SNAPSHOT on 5c4fc0a64709 with PID 9 (
/maven/resanet-catalogue-07-solution-0.0-SNAPSHOT.jar started by root in
 /)
resanet-catalogue_1    | 2016-02-03 14:51:49  DEBUG c.z.m.r.c.s.Applicat
ion:51 - Running with Spring Boot v1.3.3.RELEASE, Spring v4.2.5.RELEASE
resanet-catalogue_1    | 2016-02-03 14:51:49  INFO  c.z.m.r.c.s.Applicat
ion:670 - The following profiles are active: dev
resanet-reservations_1 | 2016-02-03 14:51:50  INFO  c.z.m.r.r.s.Applicat
ion:48 - Starting Application v0.0-SNAPSHOT on 3f3f4d1bc4ab with PID 10
(/maven/resanet-reservations-07-solution-0.0-SNAPSHOT.jar started by roo
t in /)
resanet-reservations_1 | 2016-02-03 14:51:50  DEBUG c.z.m.r.r.s.Applicat
ion:51 - Running with Spring Boot v1.3.3.RELEASE, Spring v4.2.5.RELEASE
resanet-reservations_1 | 2016-02-03 14:51:50  INFO  c.z.m.r.r.s.Applicat
ion:670 - The following profiles are active: dev
resanet-catalogue_1    | 2016-02-03 14:51:58  INFO  c.z.m.r.c.s.Applicat
ion:57 - Started Application in 8.859 seconds (JVM running for 9.242)
resanet-reservations_1 | 2016-02-03 14:52:00  INFO  c.z.m.r.r.s.Applicat
ion:57 - Started Application in 10.088 seconds (JVM running for 10.729)
resanet-catalogue_1    | 2016-02-03 14:52:02 c85b8083-eec2-4ab7-9a91-adc
5bd19bdc0 TRACE c.z.m.r.c.c.TransportController:34 - Reading all transpo
rts
```

- Nous avons appris à configurer un identifiant de corrélation dans une application spring pour des appels HTTP. Configurons maintenant la suite ELK via docker.

8.4 Configurer un elasticsearch via docker

- Nous allons utiliser le `docker-compose.yml`
- Pour cela ajouter un service nommé `elasticsearch`, son image est `elasticsearch:2.1.1`
- Monter le dossier ou il enregistre ses données sur `./data-elasticearch` en utilisant l'option volume (ie: ./data-elasticearch:/usr/share/elasticsearch/data)
- Binder le port 9200 du container avec le port 9200 de votre pc.
- Binder le port 5601 du container avec le port 5601 de votre pc (ce sera le port de kibana que l'on configurera plus tard).
- Notes :
    - Cet Elastiscsearch n'a rien à voir avec celui de l'application qui est embarqué au

sein du conteneur docker de resanet-catalog

- pour vérifier le bon fonctionnement vous pouvez utiliser `docker-compose up elasticsearch && curl localhost:9200`

8.5 Configuration de Kibana

- Pour cela ajouter un service nommé `kibana`, son image est `kibana:4.3`
- Partager le réseau entre le containeur d'elasticsearch et kibana dans le but de ne pas changer la configuration par defaut de kibana.

```
network_mode: "service:elasticsearch"
```

- Notes :
  - pour vérifier le bon fonctionnement vous pouvez utiliser `docker-compose up -d kibana && curl localhost:5601`

8.6 Configuration le Logstash

- Voici la configuration de Logstash :

```
logstash:
    image: logstash:2.1.1
    environment:
        TZ: Europe/Paris
    expose:
        - "12201"
    ports:
        - "12201:12201"
        - "12201:12201/udp"
    volumes:
        - ./gelf-conf:/conf
    links:
        - elasticsearch:elasticsearch
    command: logstash -f /conf/gelf.conf
```

- Notes :
  - le conteneur elasticsearch est lié à logstash, ce qui permet d'associer un nom d'hôte elasticsearch avec l'ip réelle du conteneur elasticsearch

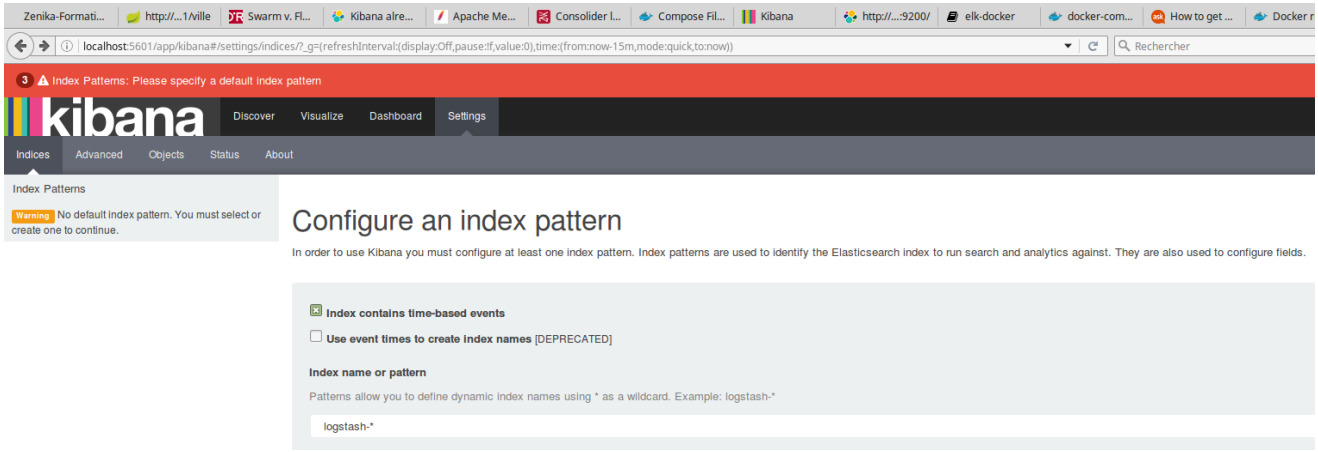
8.7 Création de la configuration de logstash

- Ecrire dans un fichier à ./gelf-conf/gelf.conf :

```
input {
  gelf {
    type => docker
    port => 12201
  }
}

output {
  elasticsearch {
    hosts => elasticsearch
  }
}
```

- A ce stade l'orque vous ouvrez un brosser et aller à `http://localhost:5601` vous devriez obtenir l'image suivante :



- Cette image indique qu'il n'y a pas encore de donnée dans Elasticsearch et Kibana ne peut détecter le mapping.

8.8 Ajouter les logs de docker à ELK

- Il faut maintenant configurer docker pour que les containeurs de resanet-reservations et resanet-catalogue loguent dans Logstash.

- Rajouter pour els services concernés dans le `docker-compose.yml` les lignes suivantes :

```
logging:
            driver: gelf
            options:
                gelf-address: "udp://localhost:12201"
```

- Puis dans Kibana à configurer le mapping sur `@timestamp`

- Lancer plusieurs requêtes sur les services resanet et explorer l'interface Kibana

remarquer que le format est le format GELF introduit dans le cours.