



Cross Cutting - Global Config Management Pattern

Global Config Management Pattern

Implementação de um Microservice que age como um servidor de configuração, onde centraliza e gerencia as propriedades e configurações dos demais Microservices da arquitetura.



Pode-se utilizar de repositórios Git ou locais para armazenar as configurações

Centraliza e gerencia as configurações dos Microservices contemplando os diferentes profiles (ex: dev, homologação, produção, etc)

Atualiza configurações e propriedades dos Microservices sem necessidade de reinicialização

Reaproveitamento das configurações globais e configurações de profile para vários Microservices da arquitetura

Configuration Hierarchy – properties/yaml

- **application.yaml:** Configurações globais da arquitetura;
 - Exemplos: Configurações de conexão ServiceRegistry, conexão com broker, etc.
- **{profile}.yaml:** Configurações de um determinado profile, aplicadas a todos Microservices deste profile;
 - Exemplos: dev.yaml, prod.yaml
- **{microservice}.yaml:** Configurações específicas de um determinado Microservice, que herda também as configurações globais;
 - Exemplos: authuser.yaml, course.yaml
- **{microservice}{profile}.yaml:** Configurações específicas de um Microservice combinadas com configurações de um determinado profile, ambos herdam também configurações globais.
 - Exemplos: authuser-dev.yaml, course-prod.yaml

Configs Endpoints - Config Server

- GET `/application/{label}`
- GET `/application/{profile}/{label}`

The screenshot shows a REST client interface with a GET request to `http://localhost:8888/application/main`. The response is a JSON object with the following structure:

```
1 {
2   "name": "application",
3   "profiles": [
4     "main"
5   ],
6   "label": null,
7   "version": "7dfde77ae926586c8dde856a5028a3544ff1492b",
8   "state": null,
9   "propertySources": [
10    {
11      "name": "https://github.com/MichelliBrito/config-server-repo/file:C:\\Users\\MICHEL~1\\AppData\\Local\\Temp\\config-repo-15682886154526007090\\application.
12      yaml",
13      "source": {
14        "eureka.client.serviceUrl.defaultZone": "http://localhost:8761/eureka",
15        "eureka.instance.hostname": "localhost"
16      }
17    }
18  ]
19 }
```

The response status is 200 OK, with a time of 450 ms and a size of 587 B. The response is saved.