THE
DEVELOPER'S
CONFERENCE

CONFIGURAÇÕES DISTRIBUÍDAS COM SPRING CLOUD CONFIG

Trilha Arquitetura Java

Emmanuel Neri

@emmanuelnerii

EMMANUEL NERI

THE DEVELOPER'S CONFERENCE

- Mestre em Desenvolvimento de Tecnologia
- Desenvolvedor (Java) desde 2010
- Professor na FACEC
- Líder técnico na Navita

CONFIGURAÇÕES

.properties

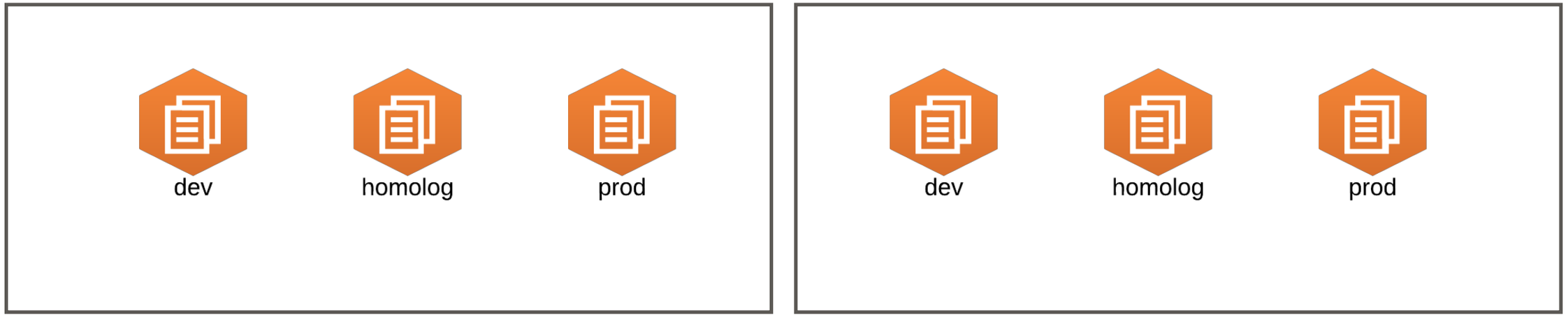
.yml

```
spring.rabbitmq.host=localhost
spring.rabbitmq.port=5672

spring.data.mongodb.host=localhost
spring.data.mongodb.port=27017


aws.accessKeyId=AKIAIMSF1DMK7EXAAAEE
aws.secretKey=wJxlrXUtnTAMX/K7MDENG
```

CONFIGURAÇÕES - UMA APLICAÇÃO

THE DEVELOPER'S CONFERENCE

CONFIGURAÇÕES - MULTI CLIENTES



Cliente X

Cliente Y

CONFIGURAÇÕES DISTRIBUÍDAS

THE DEVELOPER'S CONFERENCE

CONFIGURAÇÕES EXTERNAS

THE DEVELOPER'S CONFERENCE

SPRING CLOUD CONFIG SERVER

REPOSITÓRIO CÓDIGO FONTE

FILE STORE

BANCO DE DADOS

git

JDBC

SPRING CLOUD CONFIG

THE DEVELOPER'S CONFERENCE

- Server
  - Armazenamento das configurações
  - APIs baseadas em HTTP
  - Criptografia de configurações
    - @EnableConfigServer
- Client
  - Auto bind das configurações
  - Criptografia de configurações

SPRING CLOUD CONFIG

processor

orders

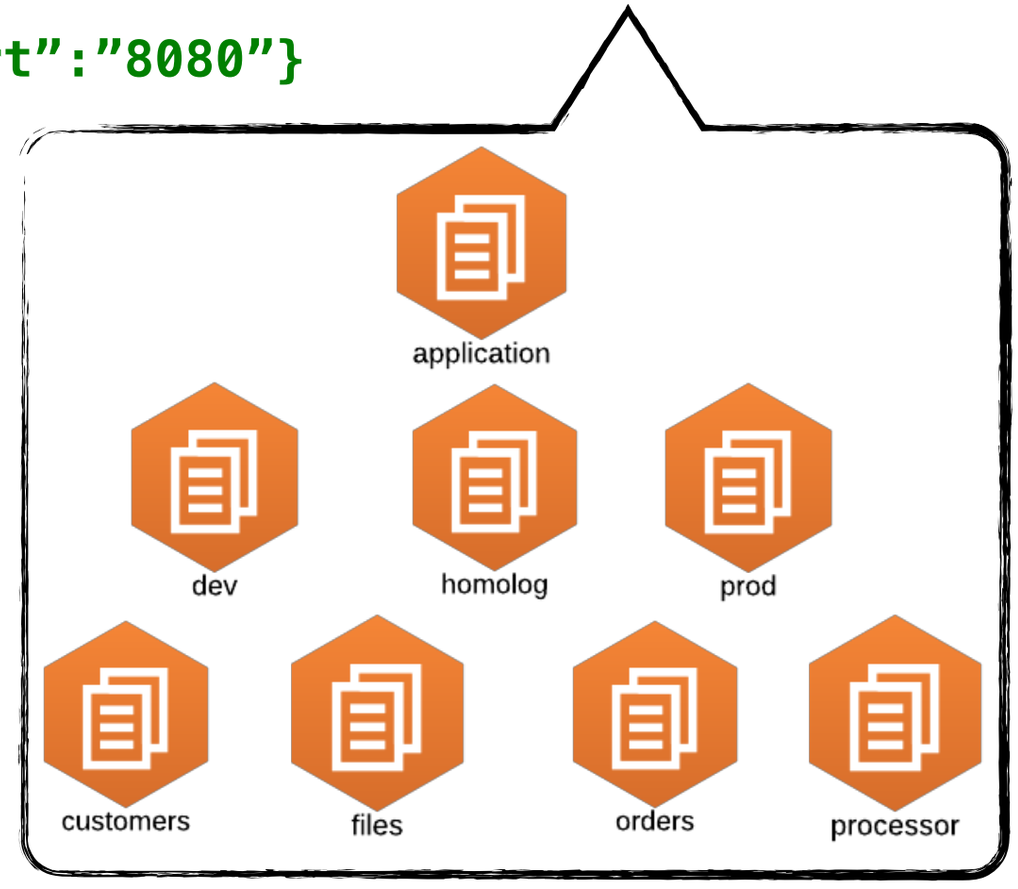
files

customers

name / profiles

{“server.port”:”8080”}

Spring Cloud Config Server



SPRING CLOUD CONFIG API

THE DEVELOPER'S CONFERENCE

- GET
  - /{application}-{profile}.properties
  - /{application}-{profile}.yml
  - /{label}/{application}-{profile}.yml
  - /{application}-{profile}.properties/{label}
  - /{application}/{profile}[/{label}]

API

http://localhost:8888/app/default

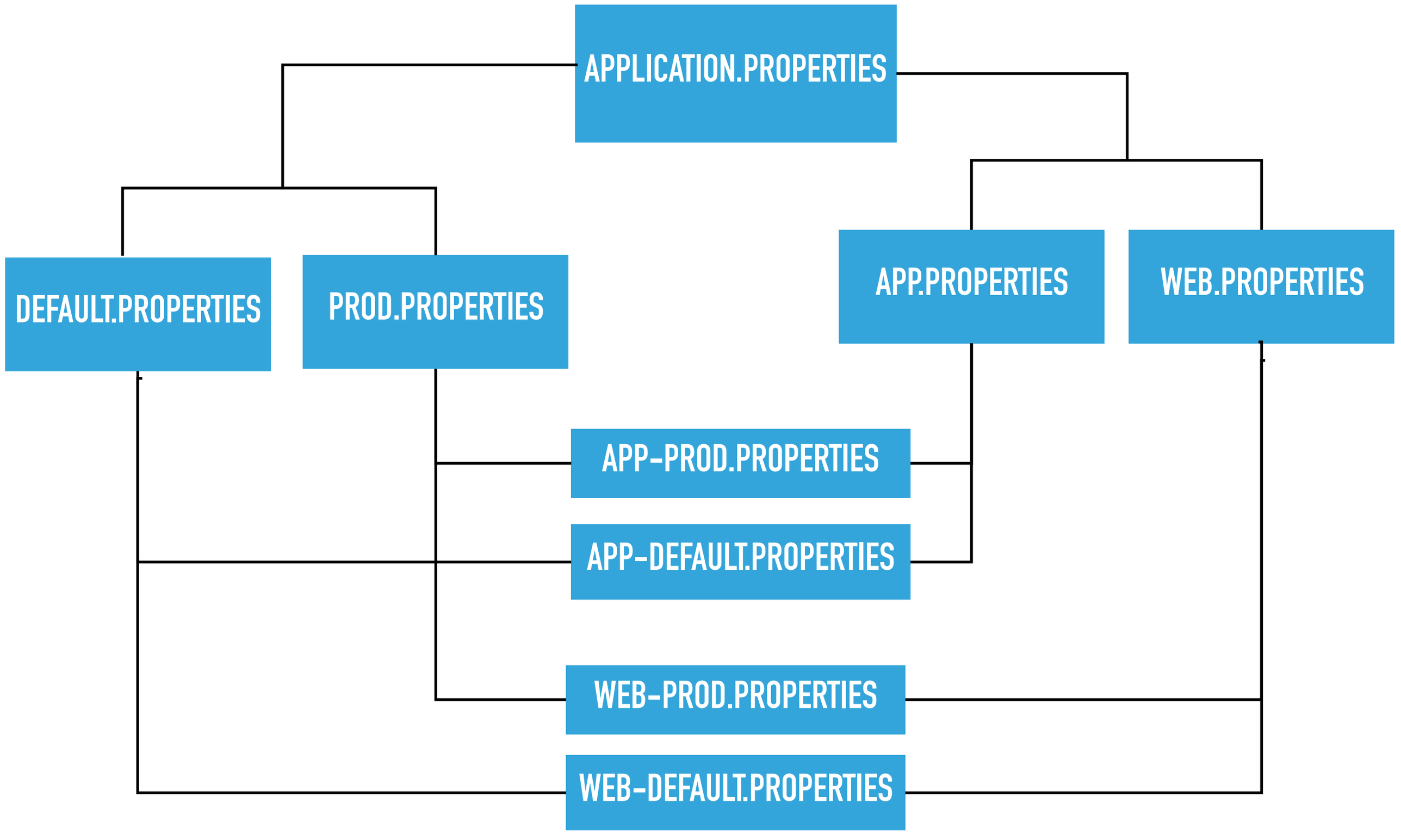

```
{
   "name":"app",
   "profiles":[
      "default"
   ],
   "label":null,
   "version":"b5f359968cf3a84e6ad7e86f52a25357927f32da"
...
```

API


```
"propertySources":[
   {
      "name":"https://github.com/emmanuelneri-blog/spring-cloud-config-configuration/app-default.properties",
      "source":{
         "hello.api.active":"false",
         "spring.security.user.name":"user",
         "spring.security.user.password":"default"
      }
   },
   {
      "name":"https://github.com/emmanuelneri-blog/spring-cloud-config-configuration/application-default.properties",
      "source":{
         "logging.level.br.com.emmanuelneri":"DEBUG"
      }
   },
   {
      "name":"https://github.com/emmanuelneri-blog/spring-cloud-config-configuration/app.properties",
      "source":{
         "server.port":"8090",
         "cliente.name":"Client I"
      }
   },
   {
      "name":"https://github.com/emmanuelneri-blog/spring-cloud-config-configuration/application.properties",
      "source":{
         spring.http.encoding.charset=UTF-8
      }
   }
]
}
```

HIERARQUIA DE CONFIGURAÇÕES

THE DEVELOPER'S CONFERENCE



CONFIGURAÇÕES

THE DEVELOPER'S CONFERENCE

Code | Issues 0 | Pull requests 0 | Projects 0 | Wiki | Insights | Settings

Configuração Spring Boot utilizadas Spring Cloud Config

spring-cloud spring-cloud-config Manage topics

24 commits | 2 branches | 0 releases | 1

Branch: master | New pull request | Create new file | Upload files | Find file

emmanuelneri Adicionando configuracao aplicacao web — Latest commit c

.gitignore	init
README.md	Create README.md
app-default.properties	atualizando senhas
app-prod.properties	atualizando senhas
app.properties	propriedades RefreshScope
application-default.properties	Ajustando properties após Spring security 5
application-prod.properties	ajustes nas configuracoes
application.properties	padronizando encoding
web-default.properties	Adicionando configuracao aplicacao web
web-prod.properties	Adicionando configuracao aplicacao web
web.properties	Adicionando configuracao aplicacao web

README.md

SPRING CLOUD CONFIG SERVER

THE DEVELOPER'S CONFERENCE

```
package br.com.emmanuelneri.config;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.cloud.config.server.EnableConfigServer;

@SpringBootApplication
@EnableConfigServer
public class CloudConfigAppConfig  {

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

}
```

```
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-config-server</artifactId>
    <version>2.0.1.RELEASE</version>
</dependency>
```

SPRING CLOUD CONFIG SERVER

THE DEVELOPER'S CONFERENCE

application.properties

```
server.port=8888

spring.application.name=configServer
spring.cloud.config.server.git.uri=https://github.com/emmanuelneri/distributed-configurations-files
```

```
spring.cloud.config.server.git.basedir
spring.cloud.config.server.git.host-key
spring.cloud.config.server.git.private-key
spring.cloud.config.server.git.known-hosts-file
spring.cloud.config.server.git.username
spring.cloud.config.server.git.password
spring.cloud.config.server.svn.uri
spring.cloud.config.server.jdbc.sql
spring.cloud.config.server.native.search-locations
```

SPRING CLOUD CONFIG CLIENT

THE DEVELOPER'S CONFERENCE

```xml
<dependency>
    <groupId>org.springframework.cloud</groupId>
    <artifactId>spring-cloud-starter-config</artifactId>
    <version>2.0.1.RELEASE</version>
</dependency>
```

bootstrap.properties

```
spring.application.name=app
spring.cloud.config.uri=http://localhost:8888
```

```
spring.cloud.config.fail-fast=true
spring.cloud.config.retry=3
```

SPRING CLOUD CONFIG CLIENT

THE DEVELOPER'S CONFERENCE

c.c.c.ConfigServicePropertySourceLocator : **Fetching config from server at : http://localhost:8888**

c.c.c.ConfigServicePropertySourceLocator : **Located environment: name=app, profiles=[app,prod]**, label=null, version=2412ba528bbbe2c1bd7d33da3a040266f497412d, state=null

b.c.PropertySourceBootstrapConfiguration : **Located property source**: CompositePropertySource {name='configService', propertySources=[MapPropertySource {name='configClient'}, MapPropertySource {name='https://github.com/emmanuelneri-blog/spring-cloud-config-configuration/**app-prod.properties**'}, MapPropertySource {name='https://github.com/emmanuelneri-blog/spring-cloud-config-configuration/**application-prod.properties**'}, MapPropertySource {name='https://github.com/emmanuelneri-blog/spring-cloud-config-configuration/**app.properties**'}, MapPropertySource {name='https://github.com/emmanuelneri-blog/spring-cloud-config-configuration/**application.properties**'}]}

SPRING CLOUD CONFIG SERVER

+

SPRING SECURITY

```
spring.cloud.config.username=config
spring.cloud.config.password=config

spring.cloud.config.token=Yc1IhlhUM7
```

CRIPTOGRAFIA

THE DEVELOPER'S CONFERENCE

- Encrypt

curl -d 'TDC-POA-2018' http://localhost:8888/encrypt

```
aefb84c0d5c5ae1fc8c40962862959947a8a8b6c8b01fb60a379fec300321598
```

- Decrypt

curl -d 'aefb84c0d5c5ae1fc8c40962862959947a8a8b6c8b01fb60a379fec300321598' http://localhost:8888/decrypt

```
TDC-POA-2018
```

```
spring.security.user.name=user
spring.security.user.password={cipher}14951724fca5177704a069ce9872301138ac1f800100703a75bff41ed84ffa86
```

CRIPTOGRAFIA - CHAVE

THE DEVELOPER'S CONFERENCE

- Key

```
encrypt.key=SpringCloudConfigEncryptSecret
```

```
aefb84c0d5c5ae1fc8c40962862959947a8a8b6c8b01fb60a379fec300321598
```

- Key store

```
encrypt.key-store.location=classpath:/distributed.configurations.jks
encrypt.key-store.password=tdcpoa2018
encrypt.key-store.alias=distributedConfigurations
encrypt.key-store.secret=tdcpoa2018
```

```
AQCaJ9DwMdOTIOmffSDW+yyX4pTxgnAGrfLA9ZOsW+nUfBXcGgMpfgdz5E7jNRQL/VA8pNf6D6Wi+eZ/
oYiIzHMdVrOcxoNx1H5FnISFC0tX2I8kLrSEcXXyes9xojyyTKuI2Va94tJThm4f872wklMFYQmW8su/
yznNTGSVgQ0B4QPj0GeM+hZ0v3wV144zqloL4HFgZo5OKjem51Z25/60+n+yeyHaLqg1/
aLKGK7SKa6+RKmRNP7bLj+xWfY0lvsGH0L1tVZkGSqQQJ3u+r33RXZsvWqFLc628/
mb7/22eK9iwJdiVhxjlSHNAIf9hiMTTUVKYZpwhTx2yEGb+VL9hohI89Hwx2o5uEFm522Y/AvQLJrn/3z1H+UCL1k7Chs=
```

POLÍTICAS

TO USE THE ENCRYPTION AND DECRYPTION FEATURES YOU NEED THE FULL-STRENGTH JCE INSTALLED IN YOUR JVM (IT IS NOT INCLUDED BY DEFAULT). YOU CAN DOWNLOAD THE "JAVA CRYPTOGRAPHY EXTENSION (JCE) UNLIMITED STRENGTH JURISDICTION POLICY FILES" FROM ORACLE AND FOLLOW THE INSTALLATION INSTRUCTIONS (ESSENTIALLY, YOU NEED TO REPLACE THE TWO POLICY FILES IN THE JRE LIB/SECURITY DIRECTORY WITH THE ONES THAT YOU DOWNLOADED).

Dockerfile

```
FROM openjdk:8-jdk-alpine
VOLUME /tmp

RUN apk upgrade --update && \
    apk add --update curl unzip && \
    curl -jksSLH "Cookie: oraclelicense=accept-securebackup-cookie" -o /tmp/
unlimited_jce_policy.zip
                  "http://download.oracle.com/otn-pub/java/jce/8/jce_policy-8.zip" && \
    unzip -jo -d ${JAVA_HOME}/jre/lib/security /tmp/unlimited_jce_policy.zip && \
    apk del curl unzip && \
    rm -rf /tmp/* /var/cache/apk/*

COPY target/server-*.jar app.jar
ENTRYPOINT ["java", "-Djava.security.egd=file:/dev/./urandom", "-jar", "/app.jar"]
```

ATUALIZAÇÃO DAS CONFIGURAÇÕES

THE DEVELOPER'S CONFERENCE

```java
@Component
@ConfigurationProperties
@RefreshScope
public class RefreshProperties {

    @Value("${cliente.name}")
    private String clientName;

    @Value("${hello.api.active}")
    private boolean helloApiActive;

}
```

curl -d {} http://localhost:8090/actuator/refresh

OUTROS

- **Serving Plain Text**
  - http://localhost:8888/web/prod/master/info.json
  - { "url": "${url}", "user": "${spring.security.user.name}" }
- **Vault**
  - spring.cloud.config.server.vault.host
- **Service Discovery**
  - Eureka / Consul
  - spring.cloud.config.discovery.enabled
- **Spring Cloud Bus**
  - spring-cloud-config-monitor
  - spring-cloud-starter-stream-rabbit
  - spring-cloud-starter-bus-amqp

EXEMPLO

THE DEVELOPER'S CONFERENCE

https://github.com/emmanuelneri/distributed-configurations

https://github.com/emmanuelneri/distributed-configurations-files

https://emmanuelneri.com.br/2018/07/09/configuracoes-distribuidas-com-spring-cloud-config/

OBRIGADO!


www.linkedin.com/in/emmanuelnerisouza


emmanuelnerisouza@gmail.com


@emmanuelnerii


www.emmanuelneri.com.br