Spring Cloud Configuration

git clone https://github.com/rugbier/SpringCloudConfig.git

¿Qué es?

Es una funcionalidad de Spring que permite crear un servidor de configuraciones para que sea atacado/accedido por microservicios para obtener las configuraciones.

También permite configurar de manera sencilla el cliente para que acceda a este microservicio

¿Como se pueden almacenar estas configuraciones?

Las configuraciones pueden guardarse como ficheros locales al servidor, ficheros remotos, repositorio git de configuraciones, etc.

Lo mas común suele ser repositorios GIT de configuraciones.

Servidor

Haciendo uso de la anotación *@SpringBootApplication* y *@EnableConfigServer*, podemos crear el servidor.

@SpringBootApplication  
@EnableConfigServer  
public class ConfigServer {  
  
 public static void main(String[] arguments) {  
 SpringApplication.*run*(ConfigServer.class, arguments);  
 }  
}

Fichero de propiedades

En el fichero de propiedades indicaremos el puerto, el repositorio git de donde se leeran las configuraciones y si se quiere hacer un clone al iniciar el servidor.

server.port=8888  
spring.cloud.config.server.git.uri=https://github.com/rugbier/config-repo  
spring.cloud.config.server.git.clone-on-start=true

pom.xml

Lo primero que tendremos que importar es el gestor de dependencias de spring cloud

<dependencyManagement>  
 <dependencies>  
 <dependency>  
 <groupId>org.springframework.cloud</groupId>  
 <artifactId>spring-cloud-dependencies</artifactId>  
 <version>${spring-cloud-dependencies.version}</version>  
 <type>pom</type>  
 <scope>import</scope>  
 </dependency>  
 </dependencies>  
</dependencyManagement>

Una vez establecido esto, configuramos las dependencias del servidor, que serán:

<dependencies>  
 <dependency>  
 <groupId>org.springframework.cloud</groupId>  
 <artifactId>spring-cloud-config-server</artifactId>  
 <version>2.2.0.RELEASE</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
 <version>2.2.2.RELEASE</version>  
 </dependency>  
</dependencies>

Propiedades del servidor

Los ficheros de propiedades que se van a servir por parte del servidor y que se encuentran en un repositorio GIT tendrán dos perfiles: development y production.

config-client-development.properties

spring.application.name=config-client  
spring.profiles.active=production  
spring.cloud.config.uri=http://localhost:8888  
spring.cloud.config.username=root  
spring.cloud.config.password=s3cr3t

config-client-production.properties

spring.application.name=config-client  
spring.profiles.active=production  
spring.cloud.config.uri=http://localhost:8888  
spring.cloud.config.username=root  
spring.cloud.config.password=s3cr3t

Cliente

Nuestro microservicio tendrá un endpoint en el que se mostrarán las propiedades cargadas desde el servidor de configuraciones. Las propiedades las cargaremos de manera tradicional con la anotación @Value

@SpringBootApplication  
@RestController  
public class ConfigClient {  
  
 @Value("${user.role}")  
 private String role;  
  
 @Value("${user.permission}")  
 private String permission;  
  
 @Value("${general.server}")  
 private String server;  
   
 public static void main(String[] args) {  
 SpringApplication.*run*(ConfigClient.class, args);  
 }  
  
 @GetMapping(value = "/whoami/{username}",  
 produces = MediaType.*TEXT\_PLAIN\_VALUE*)  
 public String whoami(@PathVariable("username") String username) {  
 return String.*format*("Hello! You're %s and you'll become a(n) %s...\nThe user permission are %s\nThe server is located in %s", username, role, permission, server);  
 }  
}

pom.xml

En la parte del cliente, tendremos las siguientes dependencias

<dependencies>  
 <dependency>  
 <groupId>org.springframework.cloud</groupId>  
 <artifactId>spring-cloud-starter-config</artifactId>  
 <version>2.2.0.RELEASE</version>  
 </dependency>  
 <dependency>  
 <groupId>org.springframework.boot</groupId>  
 <artifactId>spring-boot-starter-web</artifactId>  
 <version>2.2.2.RELEASE</version>  
 </dependency>  
</dependencies>

Fichero de propiedades

El fichero de propiedades del microservicio tiene que cargarse antes de iniciarse la aplicación.

Por lo tanto, tendremos que configurar el fichero *bootstrap.properties* con los valores para que el clienta pueda conectarse al servidor de configuraciones. En este fichero indicaremos el perfil a cargar, así como el servidor de configuraciones.

spring.application.name=config-client  
spring.profiles.active=production  
spring.cloud.config.uri=http://localhost:8888

Pruebas

Servidor

curl http://root:s3cr3t@localhost:8888/config-client/development/master  
  
{  
 "name": "config-client",  
 "profiles": [  
 "development"  
 ],  
 "label": "master",  
 "version": "1916d5aeff57980031504b6a33cd6c66c988a3c1",  
 "state": null,  
 "propertySources": [  
 {  
 "name": "https://github.com/rugbier/config-repo/config-client-development.properties",  
 "source": {  
 "'user.role": "Developer' ",  
 "'user.permission": "Restricted' ",  
 "'general.server": "http://localhost' "  
 }  
 }  
 ]  
}

Cliente

curl http://localhost:8080/whoami/Mr\_Pink  
   
Hello! You're Mr\_Pink and you'll become a(n) User' ...  
The user permission are All'  
The server is located in http://some\_value.com'