DIGITALHOUSE

Crear una aplicación en Spring Boot

Certified Tech Developer
The Ultimate Degree

¡Empecemos!

Pero comencemos por los fundamentos: vamos a agregar en el **pom.xml** la dependencia de OAuth 2.0 para que Maven haga su trabajo y nos facilite la vida:

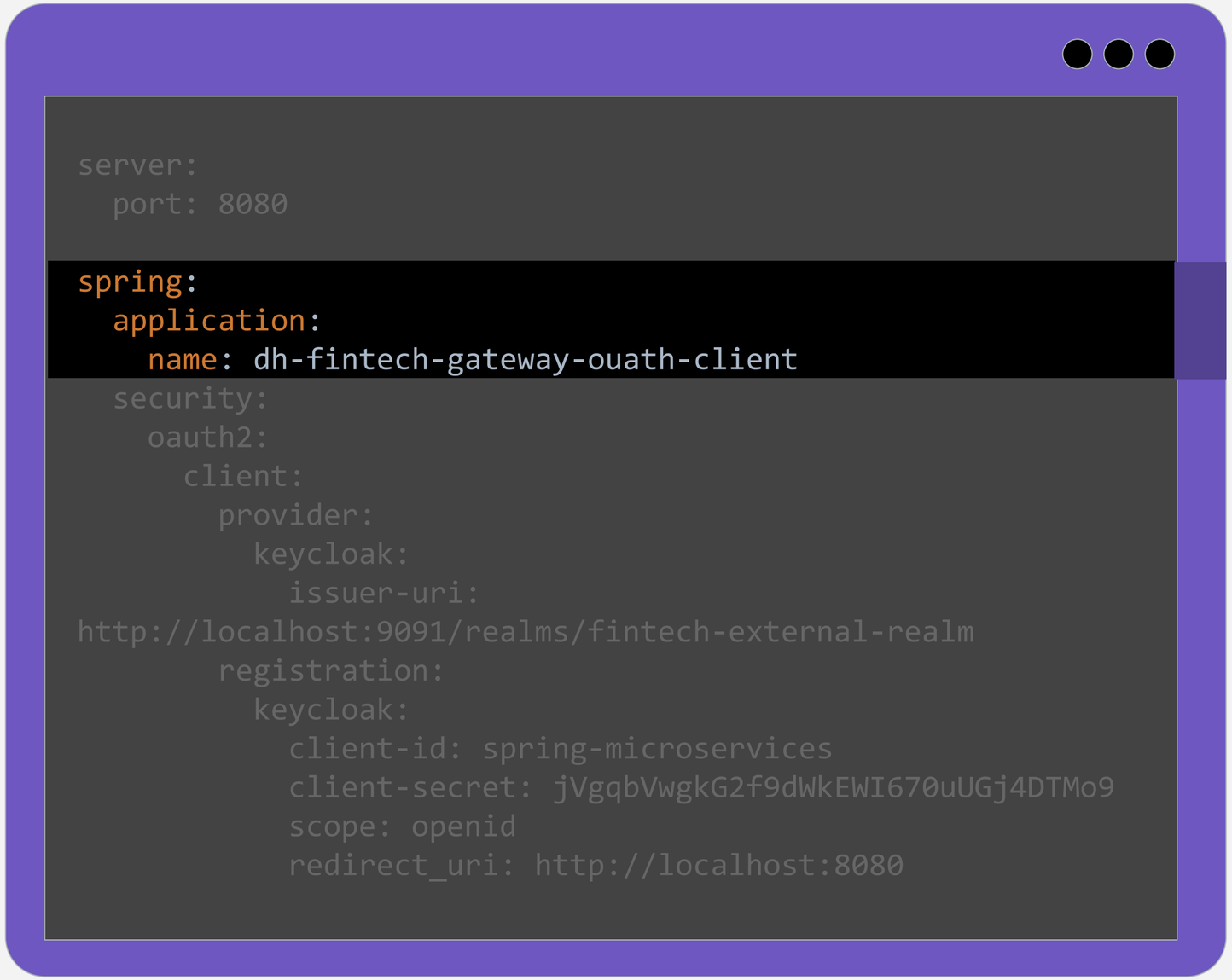
```
{}
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-oauth2-client</artifactId>
</dependency>
```

Luego, en nuestro archivo de configuración **application.yml** implementamos el uso de Keycloak para configurar la seguridad. Va un pequeño repaso:

```yaml
server:
  port: 8080
```

```yaml
spring:
  application:
    name: dh-fintech-gateway-ouath-client
  security:
    oauth2:
      client:
        provider:
          keycloak:
            issuer-uri:
http://localhost:9091/realms/fintech-external-realm
        registration:
          keycloak:
            client-id: spring-microservices
            client-secret: jVgqbVwgkG2f9dWkEWI670uUGj4DTMo9
            scope: openid
            redirect_uri: http://localhost:8080
```

Indica el puerto en el que se levanta la aplicación.



Define un nombre para nuestra aplicación.

```yaml
server:
  port: 8080

spring:
  application:
    name: dh-fintech-gateway-ouath-client
  security:
    oauth2:
      client:
        provider:
          keycloak:
            issuer-uri:
http://localhost:9091/realms/fintech-external-realm
        registration:
          keycloak:
            client-id: spring-microservices
            client-secret: jVgqbVwgkG2f9dWkEWI670uUGj4DTMo9
            scope: openid
            redirect_uri: http://localhost:8080
```

Hace la magia para que se reconozca a Keycloak como proveedor de autenticación y, como podrás ver, apunta a una URL que expone el IAM. Esta URL puede variar, pero podemos encontrarla en la URL de configuración de nuestro IAM: *http://localhost:**[puerto]**/realms/**[nombre-del-realm]**/.well-known/openid-configuration*.

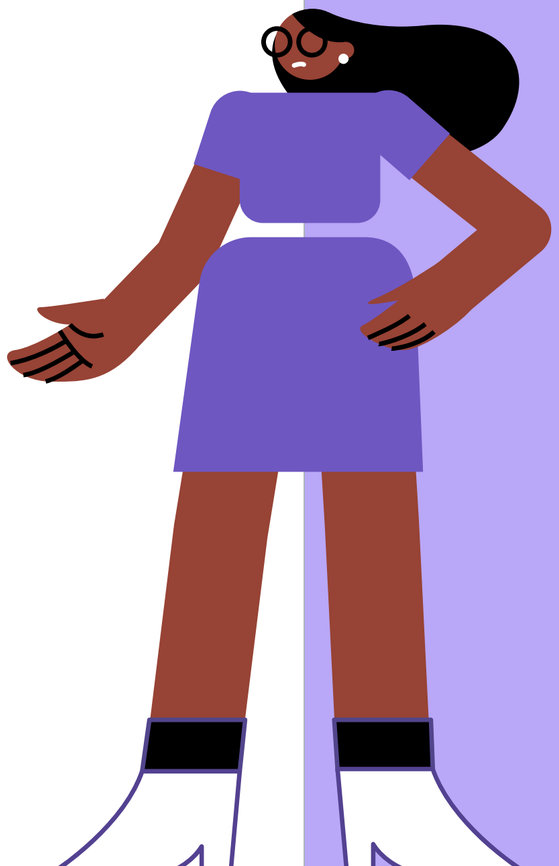
```yaml
server:
  port: 8080

spring:
  application:
    name: dh-fintech-gateway-ouath-client
  security:
    oauth2:
      client:
        provider:
          keycloak:
            issuer-uri:
http://localhost:9091/realms/fintech-external-realm
        registration:
          keycloak:
            client-id: spring-microservices
            client-secret: jVgqbVwgkG2f9dWkEWI670uUGj4DTMo9
            scope: openid
            redirect_uri: http://localhost:8080
```

client-id: Nombre cliente en Keycloak.
client-secret: Secreto del client configurado en Keycloak.
scope: Alcance de autenticación.
redirect_uri: Configuración de redirección.

Conclusiones

En este breve tutorial, comenzamos por hacer que nuestra aplicación de Spring Boot, reconozca a Keycloak como nuestro proveedor de seguridad y autenticación.

A continuación, vamos a continuar expandiendo nuestra aplicación hacia una API REST, con la cual podamos ver en funcionamiento los mecanismos que provee Keycloak.



DIGITALHOUSE

¡Muchas gracias!