1. Goals

Implement a fully working OAuth2 flow with Spring Security.

2. Lesson Notes

The “Password” Flow

We’re giving the master key (the password) to the client application. This flow will skip the */authorization* endpoint all together and directly talk to the */token* endpoint to generate an access token.

Here's a quick example interaction with the /token endpoint:

POST

http://localhost:8082/um-webapp/oauth/token?grant_...

live-test:bGl2ZS10ZXN0

And a potential response from the server:

{

"access_token": "748493f7-c17c-44c7-b4e6-e9c95dcfa511",

"token_type": "bearer",

"expires_in": 3599,

"scope": "um-webapp"

}

A quick note here is - when you interact with the API from the REST Client, in order to get to the 401 (as shown in the video) you'll need to cancel the Basic Authentication dialog prompt (which is not shown in the video).

The Live Tests and rest-assured

The flow from the POV of the client:

- first - we retrieve the access token
- then, we set that up very simply with rest-assured

Note that we're interacting with the */token*endpoint on every request and not re-using the token, so this is a potential improvement in the way we're handling the flow on the client side.

3. Resources

- Spring Security OAuth

- OAuth 2 Developers Guide