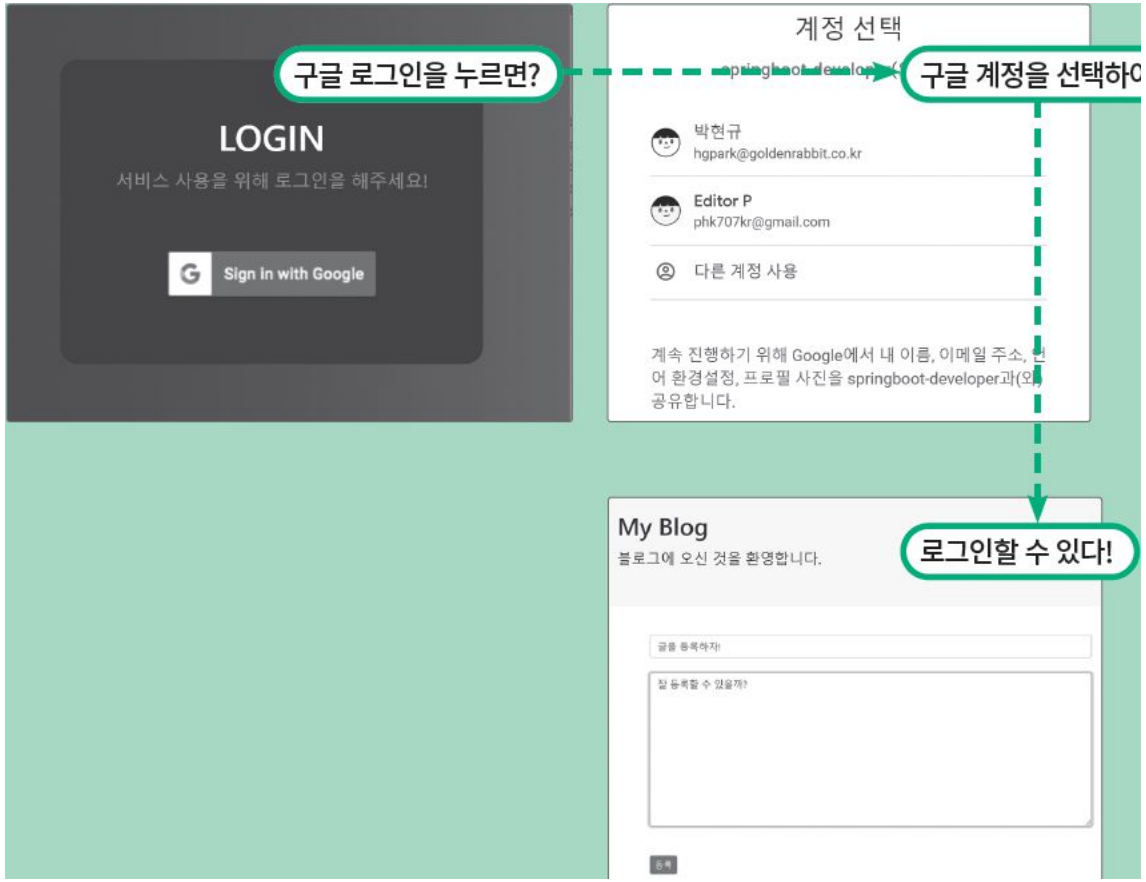
스프링 부트 3
백엔드 개발자 되기 (2판)

JPA+OAuth2+JWT+AWS와 배우는 스프링 부트 3
Java 백엔드 입문자를 위한 풀 패키지

10 OAuth2로 로그인/로그아웃 구현하기

완성 화면 미리보기


구글 로그인을 누르면?
LOGIN
서비스 사용을 위해 로그인을 해주세요!
Sign in with Google
계정 선택
구글 계정을 선택하
박현규
hgpark@goldenrabbit.co.kr
Editor P
phk707kr@gmail.com
다른 계정 사용
계속 진행하기 위해 Google에서 내 이름, 이메일 주소,
어 환경설정, 프로필 사진을 springboot-developer과(와)
공유합니다.
My Blog
블로그에 오신 것을 환영합니다.
로그인할 수 있다!

10.1 사전 지식 : OAuth

- 제 3의 서비스에 계정 관리를 맡기는 방식
  - 네이버, 구글, 페이스북, 인스타그램으로 로그인하기
- OAuth 용어

OAuth 용어 정리

· **리소스 오너**(resource owner) : 자신의 정보를 사용하도록 인증 서버에 허가하는 주체입니다. 서비스를 이용하는 사용자가 리소스 오너에 해당됩니다.

· **리소스 서버**(resource server) : 리소스 오너의 정보를 가지며, 리소스 오너의 정보를 보호하는 주체를 의미합니다. 네이버, 구글, 페이스북이 리소스 서버에 해당합니다.

· **인증 서버**(authorization server) : 클라이언트에게 리소스 오너의 정보에 접근할 수 있는 토큰을 발급하는 역할을 하는 애플리케이션을 의미합니다.

· **클라이언트 애플리케이션**(client application) : 인증 서버에게 인증을 받고 리소스 오너의 리소스를 사용하는 주체를 의미합니다. 지금 만들고 있는 서비스가 이에 해당됩니다.

10.1 사전 지식 : OAuth(cont.)

- OAuth를 사용하면 인증 서버에서 발급받은 토큰으로 리소스 서버에 리소스 오너의 정보를 요청하고 응답받아 사용할 수 있음
- 클라이언트는 다음과 같은 방법으로 리소스 오너의 정보를 획득할 수 있음
  - 본 도서는 권한 부여 코드 승인 타입을 사용함

리소스 오너 정보를 취득하는 4가지 방법

· **권한 부여 코드 승인 타입**(authorization code grant type) : OAuth 2.0에서 가장 잘 알려진 인증 방법입니다. 클라이언트가 리소스에 접근하는 데 사용하며, 권한에 접근할 수 있는 코드와 리소스 오너에 대한 액세스 토큰을 발급받는 방식입니다.

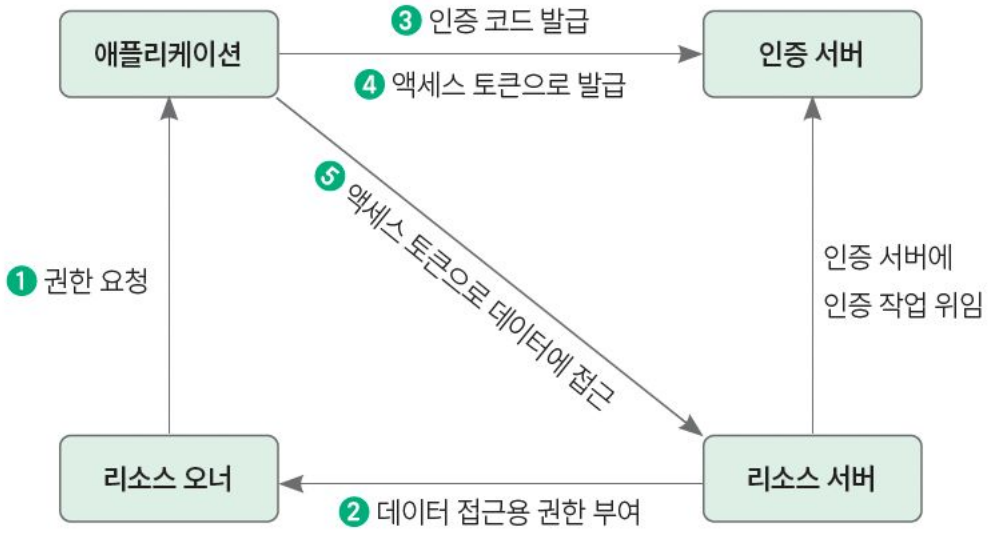
· **암시적 승인 타입**(implicit grant type) : 서버가 없는 자바스크립트 웹 애플리케이션 클라이언트에서 주로 사용하는 방법입니다. 클라이언트가 요청을 보내면 리소스 오너의 인증 과정 이외에는 권한 코드 교환 등의 별다른 인증 과정을 거치지 않고 액세스 토큰을 제공받는 방식입니다.

· **리소스 소유자 암호 자격증명 승인 타입**(resource owner password credentials) : 클라이언트의 패스워드를 이용해서 액세스 토큰에 대한 사용자의 자격 증명을 교환하는 방식입니다.

· **클라이언트 자격증명 승인 타입**(client credentials grant) : 클라이언트가 컨텍스트 외부에서 액세스 토큰을 얻어 특정 리소스에 접근을 요청할 때 사용하는 방식입니다.

10.1 사전 지식 : OAuth(cont.)

- 권한 부여 코드 승인 타입?
  - 애플리케이션, 리소스 오너, 리소스 서버, 인증 서버는 다음 순서로 인증함



10.1 사전 지식 : OAuth(cont.)

- 권한 요청
  - 스프링 부트 서버(클라이언트)가 특정 사용자 데이터에 접근하기 위해 권한 서버(구글, 카카오 등)에 요청을 보내는 것

▼ 권한 요청을 위한 파라미터 예

```
GET spring-authorization-server.example/authorize?
  client_id=66a36b4c2&
  redirect_uri=http://localhost:8080/myapp&
  response_type=code&
  scope=profile
```

10.1 사전 지식 : OAuth(cont.)

- 데이터 접근용 권한 부여
  - 인증 서버에 요청을 처음 보내면 로그인 페이지로 보내 로그인 진행 후 접근 동의 얻음(최초 1회)
  - 이후 접근 동의는 생략하고 로그인만 진행
- 인증 코드 제공
  - 로그인 성공 시 파라미터로 보낸 URL로 리다이렉션
  - 파라미터에 인증 코드 함께 제공

▼ 인증 코드 예

```
GET http://localhost:8080/myapp?code=a1s2f3mcj2
```

10.1 사전 지식 : OAuth(cont.)

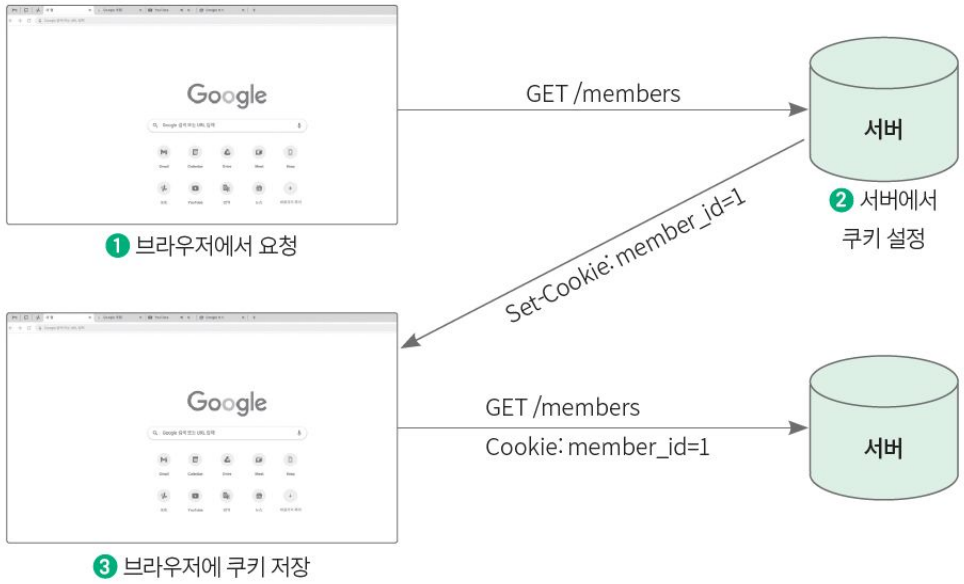
- 엑세스 토큰 응답
  - 로그인 세션에 대한 보안 자격을 증명하는 식별 코드

▼ /token POST 요청 예

```
POST spring-authorization-server.example.com/token
{
  "client_id": "66a36b4c2",
  "client_secret": "aabb11dd44",
  "redirect_uri": "http://localhost:8080/myapp",
  "grant_type": "authorization_code",
  "code": "a1b2c3d4e5f6g7h8"
}
```

10.1 사전 지식 : OAuth(cont.)

- 쿠키
  - 웹 사이트 서버에서 로컬 환경에 저장하는 작은 데이터
  - 키-값으로 구성되어 있음
  - 만료 기간, 도메인 등의 정보를 갖고 있음
  - 쿠키의 추가 과정은 다음과 같음



10.2 토큰 발급받기

- 263쪽부터 참고

10.3 스프링 시큐리티로 OAuth2 구현하고 적용하기

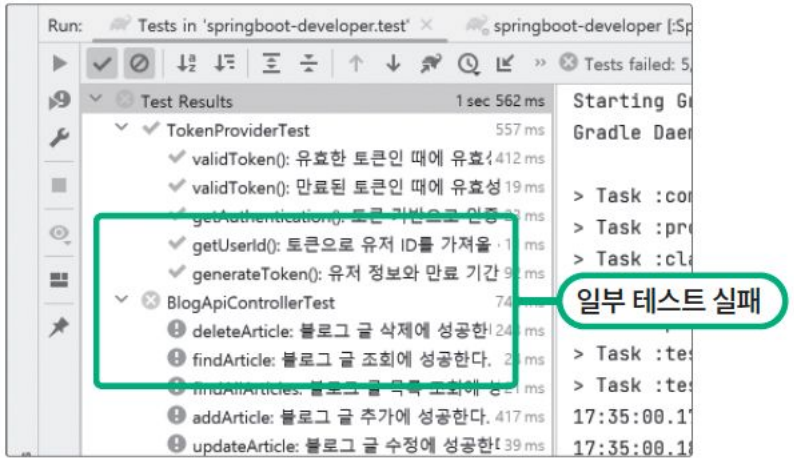
- 271쪽부터 참고
  - build.gradle
  - CookieUtil.java
  - User.java
  - OAuth2UserCustomService.java
  - /config/WebOAuthSecurityConfig.java
  - OAuth2AuthorizationRequestBasedOnCookieRepository.java
  - UserService.java
  - OAuth2SuccessHandler.java
  - Article.java
  - AddArticleRequest.java
  - BlogService.java
  - BlogApiController.java
  - ArticleViewResponse.java
  - data.sql
  - article.html
  - UserViewController.java
  - oauthLogin.html
  - token.js

10.4 OAuth2 실행 테스트하기

- 295쪽부터 참고

10.5 테스트 코드 실패 해결하고 코드 수정하기

- BlogApiControllerTest 항목이 제대로 되지 않는 문제가 있음



- 299쪽부터 참고하여 수정하면 테스트 코드 ⁄