KB금융그룹 | 국민의 평생 금융파트너

2025년 상반기 K-디지털 트레이닝

JWT의 이해

[KB] IT's Your Life

KB 국민은행

JWT(Json Web Token)이란

- **JWT (Json Web Token)**
  - JSON 포맷을 이용하여 사용자에 대한 속성을 저장하는 Claim 기반의 Web 토큰

JSON의 한 항목.
토큰에 담긴 정보하나를 클레임라 한다.

- **JWT의 구조**
  - Header.Payload.Signature 세 가지로 구성
  - 각 부분은 Base64로 인코딩되어 표현되며, 각각의 구성 요소는 . 로 구분

json으로 정보가 담겨

암호화인코딩X.

해쉬 : 큰정보 + security key -> 작은정보
서버가 securuitykey를 소유하고
작은 정보를 만들어냄

공개정보

클레임들

서명알고리즘

헤더와 페이로드정보
+
서버가 가지고있는
secretKey
=> 해쉬 정보
==서명값(과 비교)

클라가 정보+서명을 보냄
서버가 받아서 정보+securtiykey => 작은정보 생성
작은 정보와 서명을 비교. 같으면 통과. ==> 서명은 CSRF TOKEN역할을 해줌. 내가 발행한게 맞냐.

JWT(Json Web Token)이란

JWT의 구조

Header(헤더)

- alg와 typ로 구성

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

결과물에 비트수

- alg: 해싱 알고리즘. 서명(Signature) 및 토큰 검증에 사용
- typ: 토큰의 타입

JWT(Json Web Token)이란

JWT의 구조

Payload(페이로드)

- 토큰에서 사용할 정보의 조각들인 클레임(Claim)이 담겨있다.
- 클레임
  - 등록된 클레임(Registered Claim), 공개 클레임(Public Claim), 비공개 클레임(Private Claim)
  - key-value 형태

표준안 기준 / 특정사이트 기준

- 등록된 클레임(Registered Claim)
  - 토큰 정보를 표현하기 위해 이미 정해진 종류의 데이터들
  - 모두 선택적으로 작성이 가능하며 사용할 것을 권장
  - iss: 토큰 발급자(issuer)
  - sub: 토큰 제목(subject), unique한 값을 사용한다. 주로 사용자 이메일 사용
  - aud: 토큰 대상자(audience)
  - exp: 토큰 만료 시간(expiration), NumericDate 형식으로 되어 있어야 함 ex) 1480849147370
  - nbf: 토큰 활성 날짜(not before)
  - iat: 토큰 발급 시간(issued at), 토큰 발급 이후의 경과 시간
  - jti: JWT 토큰 식별자(JWT ID), 중복 방지를 위해 사용하며, 일회용 토큰(Access Token) 등에 사용

JWT(Json Web Token)이란

JWT의 구조

- Payload(페이로드)
  - 공개 클레임(Public Claim)
    - 사용자 정의 클레임
    - 공개용 정보를 위해 사용

```
{
  "https://suddiyo.tistory.com": true
}
```

  - 비공개 클레임(Private Claim)
    - 사용자 정의 클레임
    - 서버와 클라이언트 사이에 임의로 지정한 정보를 저장

```
{
  "access_token": access
}
```

JWT(Json Web Token)이란

JWT의 구조

Signature(서명)

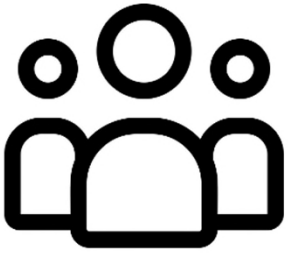
- 토큰을 인코딩하거나 유효성 검증을 할 때 사용하는 고유한 암호화 코드
- 서명은 헤더와 페이로드, 그리고 비밀 키를 기반으로 생성
- 해당 토큰이 변조되지 않았음을 확인하기 위한 메커니즘

- [서명 생성 과정]
  1. 헤더(Header)와 페이로드(Payload)의 값을 각각 BASE64로 인코딩
  2. 인코딩한 값을 비밀 키를 이용해 헤더(Header)에서 정의한 알고리즘으로 해싱
  3. 해싱한 값을 다시 BASE64로 인코딩하여 생성

Spring Security + JWT 동작 원리

로그인

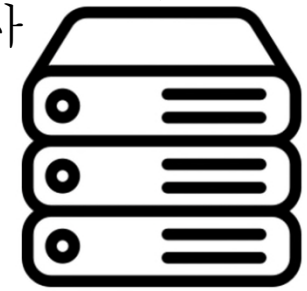
equals메소드말고
matches메소드로
검사



클라이언트

1. ID/PW 로 로그인 요청
2. 검증 후 Access Token + Refresh Token 발급
3. API 요청 시 헤더에 Access Token 담아서 요청

에다가 요청한 유저네임을
보고 동작.



서버

페이로드에 유저네임,
권한정보. 공개해도
괜찮은 정보 넣고.
헤더, 페이로드로부터
jwt1==인증억세스토큰
jwt2==리프레시토큰
2개를 만들어

1. 클라이언트에서 서버로 ID/PW로 로그인을 요청
2. 서버에서 검증 과정을 거쳐 해당 유저가 존재하면, Access Token + Refresh Token 을 발급
3. 클라이언트는 요청 헤더에 2번에서 발급받은 Access Token 을 포함하여 API를 요청
4. 해당 토큰의 유효시간(만기)을 확인함.
5. 아직 만기 안됐으면 서명확인하고 요청 동작, 만기됐다면 갱신과정

access토큰은 유지기간 짧음

refresh토큰은 만기기간 길게

클라이언트(뷰) 입장에서는 access token과 refresh token어디서 관린할까 pinia에서 관리한다.
새로고침으로 인한 날라가는 것을 방지하기 위해서
브라우저의 localStroage나 cookie에 저장한다.

Spring Security + JWT 동작 원리

Access Token

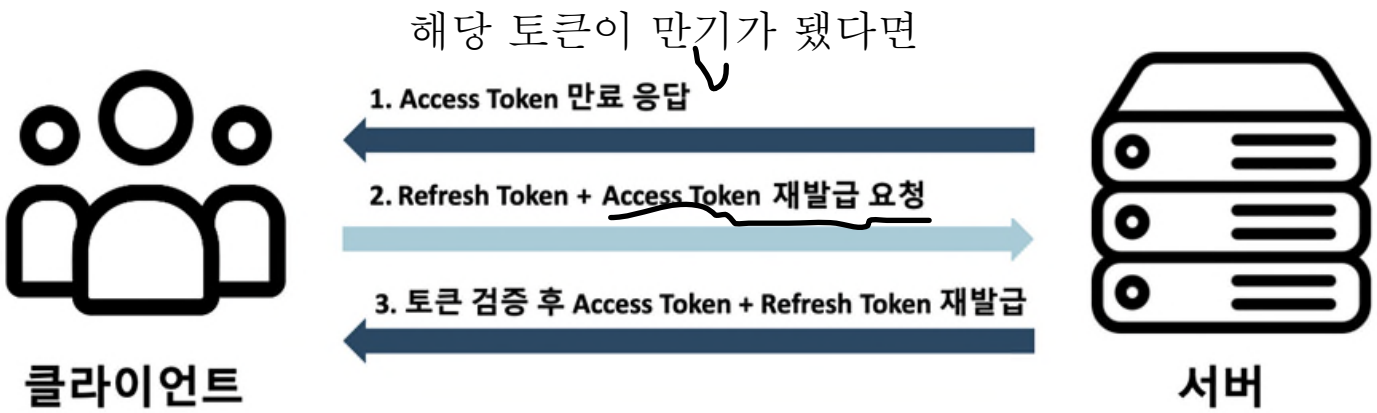
- 인증된 사용자가 특정 리소스에 접근할 때 사용되는 토큰
  - 클라이언트는 Access Token을 사용하여 인증된 사용자의 신원을 확인하고, 서비스 또는 리소스에 접근
  - 유효 기간이 지나면 만료 (expired)
  - 만료된 경우, 새로운 Access Token을 얻기 위해 Refresh Token 사용

Refresh Token

- Access Token의 갱신을 위해 사용되는 토큰
  - 일반적으로 Access Token과 함께 발급
  - Access Token이 만료되면 Refresh Token을 사용하여 새로운 Access Token 발급
  - 사용자가 지속적으로 인증 상태를 유지할 수 있도록 도와줌 (매번 로그인 다시 하지 않아도 됨)
  - 보안 상의 이유로 Access Token보다 긴 유효 기간 가짐

Spring Security + JWT 동작 원리

토큰 갱신 Refresh



1. 클라이언트 Access Token으로 요청을 보냈으나, Access Token의 유효 시간 만료
2. Refresh Token이용하여 Access Token 재발급 요청
3. 클라이언트는 요청 헤더에 재발급받은 Access Token을 포함하여 API를 다시 요청

→ 위 과정은 사용자가 모르게 자동으로 처리되야 함(axios 인터셉터에서 처리)

4. 새로운 엑세스 토큰을 통해서 요청