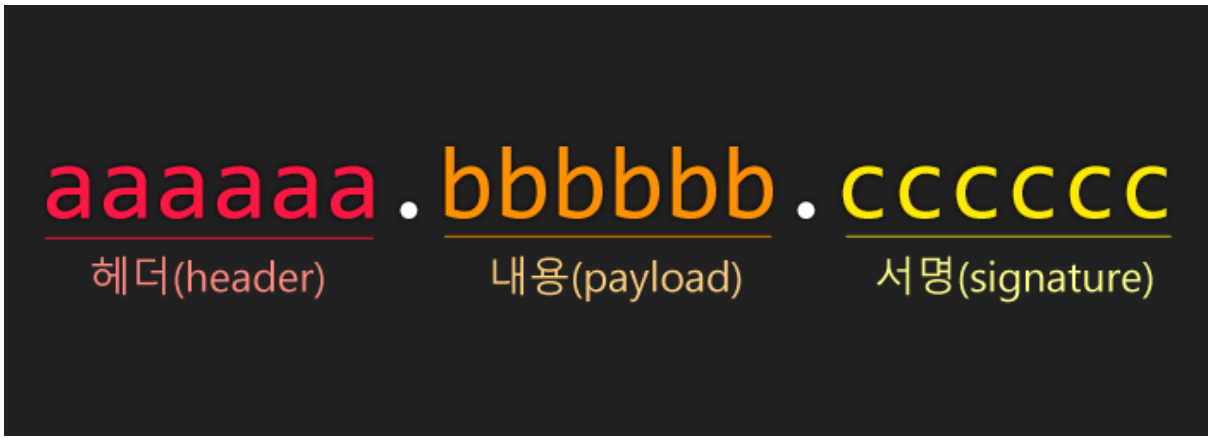
# JWT 토큰

| | |
|---|---|
| 태그 | |
| 작성자 | 조성찬 |

## JWT 인증 & Spring Security




### Header

- type : 토큰의 타입 지정. JWT 토큰의 경우 `JWT` 로 고정
- alg : 해싱 알고리즘. 보통 `HMAC SHA256` 혹은 `RSA` 를 사용한다. 이 알고리즘은 토큰을 검증할 때 사용하는 signature 부분에서 사용된다.
    - signature 부분을 가지고 어떻게 토큰을 검증하는가?

이 정보를 하나로 묶어서 `base64` 로 인코딩한다.

### Payload

토큰에 담을 정보를 넣는다. 정보 한 조각을 `claim` 이라고 부르고 name : value 한 쌍으로 이루어져 있다. 토큰 안의 Payload 안에는 여러 claim을 넣을 수 있다.

claim의 종류는 다음과 같이 3종류로 나눌 수 있다.

- 등록된 (registered) 클레임

- registered 클레임은 모두 선택적이다. 굳이 넣지 않아도 된다.

  - `iss` : 토큰 발급자 (issuer)
  - `sub` : 토큰 제목 (subject)
  - `aud` : 토큰 대상자 (audience)
  - `exp` : 토큰의 만료시간 (expiraton), 시간은 NumericDate 형식으로 되어있어야 하며 (예: 1480849147370) 언제나 현재 시간보다 이후로 설정되어있어야합니다.
  - `nbf` : Not Before 를 의미하며, 토큰의 활성 날짜와 비슷한 개념입니다. 여기에도 NumericDate 형식으로 날짜를 지정하며, 이 날짜가 지나기 전까지는 토큰이 처리되지 않습니다.
  - `iat` : 토큰이 발급된 시간 (issued at), 이 값을 사용하여 토큰의 `age` 가 얼마나 되었는지 판단 할 수 있습니다.
  - `jti` : JWT의 고유 식별자로서, 주로 중복적인 처리를 방지하기 위하여 사용됩니다. 일회용 토큰에 사용하면 유용합니다.

- 공개 (public) 클레임
  - 충돌이 방지된 (collision resistant) 이름을 가지고 있어야 한다. 충돌을 방지하기 위해서는 클레임 이름을 URI 형식으로 짓는다.

```
{
    "https://velopert.com/jwt_claims/is_admin": true
}
```

- 비공개 (private) 클레임
  - 양 측간 (클라이언트 ↔ 서버) 협의 하에 사용되는 클레임
  - 이름이 중복되어 충돌이 될 수 있으니 사용될 때에 유의해야 한다.

# 예제 Payload

```
{
    "iss": "velopert.com",
    "exp": "1485270000000",
    "https://velopert.com/jwt_claims/is_admin": true,
    "userId": "11028373727102",
    "username": "velopert"
}
```

헤더의 인코딩(base64) 값과 정보의 인코딩(base64) 값을 합친 후, 주어진 비밀키로 해싱한다.

> *실제로 JWT 를 서비스에서 사용할때는 우리가 직접 base64 인코딩을 하거나 SHA256 해싱을 할 일은 없습니다. JWT 담당 라이브러리에 설정만 해주면 자동으로 손쉽게 만들고, 또 검증도 쉽게해주기 때문이지요.*
>
> - 하지만 기능 구현이 어떻게 되어 있는지는 알아야 한다. 원리를 이해해야 한다.

[JWT] JSON Web Token 소개 및 구조

지난 포스트에서는 토큰 기반 인증 시스템의 기본적인 개념에 대하여 알아보았습니다. 이 포스트를 읽기 전에, 토큰 기반 인증 시스템에 대해서 잘 모르시는 분들은 지난 포스트를 꼭 읽어주세요. 이번 포스트에

https://velopert.com/2389