


JWT in java

<https://github.com/jwt/jjwt>

העתקנו את כל התלויות האלו

 README.md

JDK Projects

If you're building a (non-Android) JDK project, you will want to define the following dependencies:

Maven

```
<dependency>
  <groupId>io.jsonwebtoken</groupId>
  <artifactId>jjwt-api</artifactId>
  <version>0.11.2</version>
</dependency>
<dependency>
  <groupId>io.jsonwebtoken</groupId>
  <artifactId>jjwt-impl</artifactId>
  <version>0.11.2</version>
  <scope>runtime</scope>
</dependency>
<dependency>
  <groupId>io.jsonwebtoken</groupId>
  <artifactId>jjwt-jackson</artifactId> <!-- or jjwt-gson if Gson is preferred -->
  <version>0.11.2</version>
  <scope>runtime</scope>
</dependency>
<!-- Uncomment this next dependency if you are using JDK 10 or earlier and you also want to use
      RSASSA-PSS (PS256, PS384, PS512) algorithms.  JDK 11 or later does not require it for those algorithms:
<dependency>
  <groupId>org.bouncycastle</groupId>
  <artifactId>bcprov-jdk15on</artifactId>
  <version>1.60</version>
  <scope>runtime</scope>
</dependency>
-->
```

יצרנו תיקייה חדשה בשם jwt שהיא תדאג לנו לכל האימותים

יצרנו מחלקה חדשה שהיא תטפל בלקבל את השם משתמש והסיסמא מהבקשה

```
public class UsernameAndPasswordAuthenticationRequest {
    private String username;
    private String password;

    public UsernameAndPasswordAuthenticationRequest() {
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }
}
```

ועוד מחלקה שתוודא את הטוקן עצמו

```
/**
 * Validate JWT tokens
 */
public class JwtUsernameAndPasswordAuthenticationFilter extends UsernameAndPasswordAuthenticationFilter {
```

חלק ראשון אימות האישורים

מתודה שתוודא את הcredential שהיזר שולח

יצרנו משתנה של המחלקה שהוא ינהל את האותנטיקציה שלנו (אחרי זה נטפל בשגיאה, הוא צריך הזרקת תלות).

אחרי זה קיבלנו את השם משתמש והסיסמא מהבקשה ותוך האובייקט מסוג המחלקה שיצרנו קודם

עכשיו כדי לבצע אותנטיקציה לשם משתמש והסיסמא אנחנו צריכים לשים אותם בתוך אובייקט שמממש את Authentication

ואז ביצענו לזה אותנטיקציה

```
private final AuthenticationManager authenticationManager;

@Override
public Authentication attemptAuthentication(HttpServletRequest request,
                                           HttpServletResponse response) throws AuthenticationException {
    try {
        UsernameAndPasswordAuthenticationRequest authenticationRequest = new ObjectMapper()
            .readValue(request.getInputStream(), UsernameAndPasswordAuthenticationRequest.class);

        Authentication authentication = new UsernamePasswordAuthenticationToken(
            authenticationRequest.getUsername(),
            authenticationRequest.getPassword()
        );

        Authentication authenticate = authenticationManager.authenticate(authentication);
        return authenticate;
    } catch (IOException e) {
        throw new RuntimeException(e);
    }
}
```

חלק שני יצירה של טוקן ושליחה שלו

בחלק הזה בוא נדרוס את המתודה successfulAuthentication, היא תקרה רק אם המתודה attemptAuthentication עברה בהצלחה

```
@Override
protected void successfulAuthentication(HttpServletRequest request,
                                         HttpServletResponse response,
                                         FilterChain chain, Authentication authResult
) throws IOException, ServletException {
    super.successfulAuthentication(request, response, chain, authResult);
}
```

יצרנו מפתח שנוכל להגן על ה signature

ואז השתמשנו ב Jwts כדי ליצור את הטוקן, ואז שמנו בתוך header של התשובה את הטוקן

```
@Override
protected void successfulAuthentication(HttpServletRequest request,
                                         HttpServletResponse response,
                                         FilterChain chain, Authentication authResult
) throws IOException, ServletException {
    String key = "secure_secure_secure_secure_secure_";
    String token = Jwts.builder()
        .setSubject(authResult.getName()) // השם משתמש
        .claim(s: "authorities", authResult.getAuthorities()) // ההרשאות שלו
        .setIssuedAt(new java.util.Date()) // מתי זה קרה
        .setExpiration(Date.valueOf(LocalDate.now().plusWeeks(2))) // מתי זה נגמר
        .signWith(Keys.hmacShaKeyFor(key.getBytes())) // ה signature עם מפתח
        .compact(); // יצירה שלו

    response.addHeader(s: "Authorization", s1: "Bearer " + token);
}
```