KB금융그룹 | 국민의 평생 금융파트너

2025년 상반기 K-디지털 트레이닝

Spring Security JWT 인증

[KB] IT's Your Life

KB 국민은행

Spring Security JWT 인증

- **다음과 같이 프로젝트를 준비하세요.**
  - secserver 프로젝트를 apiserver로 복사
    - settings.gradle에서 프로젝트명 apiserver로 변경

Spring Security JWT 인증

- **Api 서버를 위한 기본 security 설정을 SecurityConfig.java에 추가하세요.**
  - PasswordEncorder 빈 등록
  - UserDetailsService 빈 등록
  - AuthenticationManager 빈 등록

  - 한글 인코딩 필터 설정
  - CORS(Cross Origin Resource Sharing) 허용
  - csrf 기능 비활성화
  - formLogin 기능 비활성화
  - session의 생성 모드를 stateless 모드로 설정

security.config.SecurityConfig.java

```
package org.scoula.security.config;
…
import org.springframework.web.cors.UrlBasedCorsConfigurationSource;
…

@Configuration
@EnableWebSecurity
@Log4j2
@MapperScan(basePackages  = {"org.scoula.security.account.mapper"})
@ComponentScan(basePackages  = {"org.scoula.security"})
@RequiredArgsConstructor
public class SecurityConfig extends WebSecurityConfigurerAdapter {

    private final UserDetailsService userDetailsService;
```

security.config.SecurityConfig.java

```
// 문자셋 필터
public CharacterEncodingFilter encodingFilter() {
    CharacterEncodingFilter encodingFilter = new CharacterEncodingFilter();
    encodingFilter.setEncoding("UTF-8");
    encodingFilter.setForceEncoding(true);
    return encodingFilter;
}

@Bean
public PasswordEncoder passwordEncoder() {
    return new BCryptPasswordEncoder();
}

// AuthenticationManager 빈 등록
@Bean
public AuthenticationManager authenticationManager() throws Exception {
    return super.authenticationManager();
}
```

Spring Security JWT 인증

security.config.SecurityConfig.java

```
// cross origin 접근 허용
@Bean
public CorsFilter corsFilter() {
    UrlBasedCorsConfigurationSource source = new UrlBasedCorsConfigurationSource();
    CorsConfiguration config = new CorsConfiguration();
    config.setAllowCredentials(true);
    config.addAllowedOriginPattern("*");
    config.addAllowedHeader("*");
    config.addAllowedMethod("*");
    source.registerCorsConfiguration("/**", config);
    return new CorsFilter(source);
}

// 접근 제한 무시 경로 설정 – resource
@Override
public void configure(WebSecurity web) throws Exception {
    web.ignoring().antMatchers("/assets/**", "/*", "/api/member/**");
}
```

Spring Security JWT 인증

security.config.SecurityConfig.java

```
@Override
public void configure(HttpSecurity http) throws Exception {
    // 한글 인코딩 필터 설정
    http.addFilterBefore(encodingFilter(), CsrfFilter.class);

    http.httpBasic().disable()                          // 기본 HTTP 인증 비활성화
        .csrf().disable()      // CSRF 비활성화
        .formLogin().disable()  // formLogin 비활성화     관련 필터 해제
        .sessionManagement().sessionCreationPolicy(SessionCreationPolicy.STATELESS); // 세션 생성 모드 설정
}

// Authentication Manger 구성
@Override
protected void configure(AuthenticationManagerBuilder auth) throws Exception {
    auth
      .userDetailsService(userDetailsService)
      .passwordEncoder(passwordEncoder());
}
}
```

Spring Security JWT 인증

- JWT 자바 라이브러리 의존성을 추가하세요.

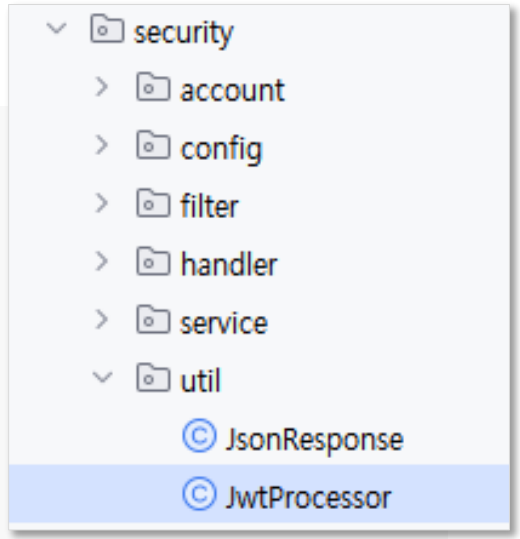
build.gradle

```
implementation("io.jsonwebtoken:jjwt-api:0.11.5")
runtimeOnly("io.jsonwebtoken:jjwt-impl:0.11.5")
implementation("io.jsonwebtoken:jjwt-jackson:0.11.5")
```

Spring Security JWT 인증

- org.scoula.security.util 패키지에 Jwt처리 유틸리티 클래스 JwtProcessor를 정의하세요.

security.util.JwtProcessor.java




```
package org.scoula.security.util;

import io.jsonwebtoken.Claims;
import io.jsonwebtoken.Jws;
import io.jsonwebtoken.Jwts;
import io.jsonwebtoken.security.Keys;
import org.springframework.stereotype.Component;

import java.nio.charset.StandardCharsets;
import java.security.Key;
import java.util.Date;

@Component
public class JwtProcessor {
    static private final long TOKEN_VALID_MILISECOND = 1000L * 60 * 5; // 5 분

    private String secretKey                    = "충분히 긴 임의의(랜덤한) 비밀키 문자열 배정 ";
    private Key key = Keys.hmacShaKeyFor(secretKey.getBytes(StandardCharsets.UTF_8));

//    private Key key = Keys.secretKeyFor(SignatureAlgorithm.HS256);  -- 운영시 사용
```

security.util.JwtProcessor.java

```
// JWT 생성
public String generateToken(String subject) {
    return Jwts.builder()
            .setSubject(subject)
            .setIssuedAt(new Date())
            .setExpiration(new Date(new Date().getTime() + TOKEN_VALID_MILISECOND))
            .signWith(key)
            .compact();
}

// JWT Subject(username) 추출 - 해석 불가인 경우 예외 발생
// 예외 ExpiredJwtException, UnsupportedJwtException, MalformedJwtException, SignatureException,
//      IllegalArgumentException
public String getUsername(String token) {
    return Jwts.parserBuilder()
            .setSigningKey(key)
            .build()
            .parseClaimsJws(token)
            .getBody()
            .getSubject();
}
```

security.util.JwtProcessor.java

```
  // JWT 검증(유효 기간 검증) - 해석 불가인 경우 예외 발생
  public boolean validateToken(String token) {
    Jws<Claims> claims = Jwts.parserBuilder()
        .setSigningKey(key)
        .build()
        .parseClaimsJws(token);
    return true;
  }
}
```

Spring Security JWT 인증

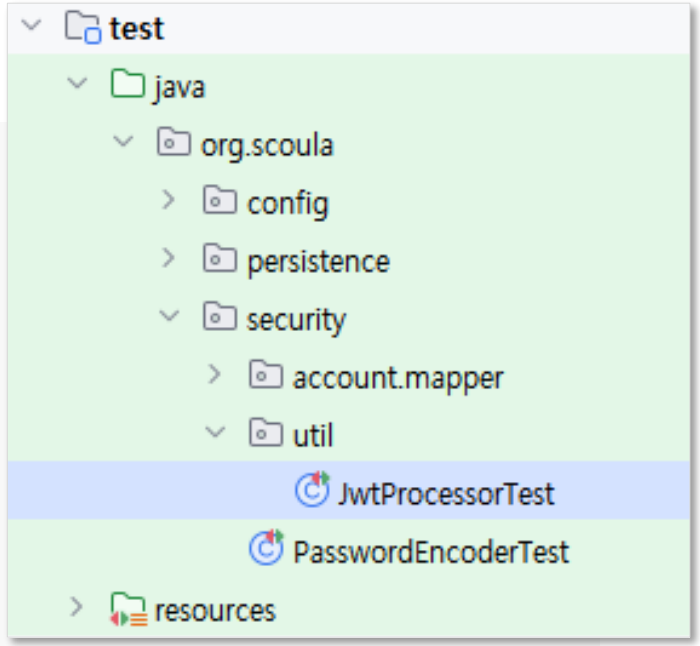
- **JwtProcessorTest 테스트 클래스를 정의하고, 다음을 테스트 하세요.**
  - 특정 사용자 ID에 대해서 JWT 생성
  - 생성된 JWT에서 사용자 ID 추출
  - 생성된 JWT 검증

Spring Security JWT 인증

test :: JwtProcessorTest.java

```
@ExtendWith(SpringExtension.class)
@ContextConfiguration(classes = { RootConfig.class, SecurityConfig.class })
@Log4j2
class JwtProcessorTest {
   @Autowired
   JwtProcessor jwtProcessor;

   @Test
   void generateToken() {
      String username = "user0";
      String token = jwtProcessor.generateToken(username);
      log.info(token);
      assertNotNull(token);
   }
```



INFO : org.scoula.security.util.JwtProcessorTest - **eyJhbGciOiJIUzM4NCJ9.eyJzdWIiOiJ1c2VyMCIsImlhdCI6MTcyMTgwMjc4NCwiZXhwIjoxNzIxODAzMDg0fQ.nwD4rIroYL6hr_-Esav8KIsHw573MbAiTT-Nz_yYHI8bMcyGZMOEjMt0Own3io_c**

복사하여 다음 테스트에 사용

Spring Security JWT 인증

test :: JwtProcessorTest.java

```
@Test
void getUsername() {
    String token = "eyJhbGciOiJIUzM4NCJ9.eyJzdWIiOiJ1c2VyMCIsImlhdCI6MTcyMTgwMjc4NCwiZXhwIjoxNzIxODAzMDg0fQ.nwD4rIroYL6hr_-Esav8KIsHw573MbAiTT-Nz_yYHI8bMcyGZMOEjMt0Own3io_c";

    String username = jwtProcessor.getUsername(token);
    log.info(username);
    assertNotNull(username);
}
```

INFO : org.scoula.security.util.JwtProcessorTest - **user0**

5분전이면 성공,

Spring Security JWT 인증

test :: JwtProcessorTest.java

```
    @Test
    void validateToken() {
        // 5분 경과 후 테스트
        String token = "eyJhbGciOiJIUzM4NCJ9.eyJzdWIiOiJ1c2VyMCIsImlhdCI6MTcyMTgwMjc4NCwiZXhwIjoxNzIxODAzMDg0fQ.nwD4rIroYL6hr_-Esav8KIsHw573MbAiTT-Nz_yYHI8bMcyGZMOEjMt0Own3io_c";

        boolean isValid = jwtProcessor.validateToken(token); // 5분 경과 후면 예외 발생
        log.info(isValid);
        assertTrue(isValid);    // 5분전이면 true
    }
}
```

JWT expired at 2024-07-24T06:38:04Z. Current time: 2024-07-24T06:38:45Z, a difference of 41480 milliseconds. Allowed clock skew: 0 milliseconds.
io.jsonwebtoken.ExpiredJwtException: JWT expired at 2024-07-24T06:38:04Z. ...