HOW CREATE JWT

# Add dependencies

<!-- io.jsonwebtoken -->

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt-api</artifactId>

<version>0.10.7</version>

</dependency>

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt-impl</artifactId>

<version>0.10.7</version>

<scope>runtime</scope>

</dependency>

<dependency>

<groupId>io.jsonwebtoken</groupId>

<artifactId>jjwt-jackson</artifactId>

<version>0.10.7</version>

<scope>runtime</scope>

</dependency>

# Create Configuration POJO

public class Configuracion {

private String user;

private String password;

private String environment;

private String role;

private Integer sessionId;

private long tokenExpiration;

public Configuracion(String user, String password, String environment, String role, Integer sessionId, long tokenExpiration) {

this.user = user;

this.password = password;

this.environment = environment;

this.role = role;

this.sessionId = sessionId;

this.tokenExpiration = tokenExpiration;

}

public Configuracion() {

super();

}

public String getUser() {

return user;

}

public void setUser(String user) {

this.user = user;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getEnvironment() {

return environment;

}

public void setEnvironment(String environment) {

this.environment = environment;

}

public String getRole() {

return role;

}

public void setRole(String role) {

this.role = role;

}

public Integer getSessionId() {

return sessionId;

}

public int getSessionIdAsInt() {

return Integer.parseInt(sessionId.toString());

}

public void setSessionId(Integer sessionId) {

this.sessionId = sessionId;

}

public long getTokenExpiration() {

return tokenExpiration;

}

public void setTokenExpiration(long tokenExpiration) {

this.tokenExpiration = tokenExpiration;

}

# Create JWT Routine

import io.jsonwebtoken.JwtBuilder;

import io.jsonwebtoken.Jwts;

import io.jsonwebtoken.SignatureAlgorithm;

import java.security.Key;

import javax.crypto.spec.SecretKeySpec;

import javax.xml.bind.DatatypeConverter;

private String getJWT(Configuracion config) {

long ttlMillis = config.getTokenExpiration();

//The JWT signature algorithm we will be using to sign the token

SignatureAlgorithm signatureAlgorithm = SignatureAlgorithm.HS256;

long nowMillis = System.currentTimeMillis();

Date now = new Date(nowMillis);

//We will sign our JWT with our ApiKey secret

byte[] apiKeySecretBytes = DatatypeConverter.parseBase64Binary("123456789012345678901234567890123456789012345678901234567890");

Key signingKey = new SecretKeySpec(apiKeySecretBytes, signatureAlgorithm.getJcaName());

//Let's set the JWT Claims

JwtBuilder builder = Jwts.builder().setId("1231231")

.setIssuedAt(now)

.setSubject("Subject")

.setIssuer("Issue")

.claim("user", config.getUser())

.claim("password", config.getPassword())

.claim("environment", config.getEnvironment())

.claim("role", config.getRole())

.claim("sessionId", config.getSessionId())

.signWith(signingKey, signatureAlgorithm);

//if it has been specified, let's add the expiration

if (ttlMillis > 0) {

long expMillis = nowMillis + ttlMillis;

Date exp = new Date(expMillis);

builder.setExpiration(exp);

}

return builder.compact();

}

# How use it

int sessionID = 0;

Configuracion config = new Configuracion();

config.setUser(configuracion.getUser());

config.setPassword(configuracion.getPassword());

config.setEnvironment(configuracion.getEnvironment());

config.setRole(configuracion.getRole());

config.setSessionId((int) sessionID);

config.setTokenExpiration(0L);

String token = getJWT(config);