Biblioteca JWT de Java

Para usar la biblioteca `com.nimbusds.jwt` en Java para extraer información como la fecha de expiración, la fecha de emisión y otros reclamos de un JSON Web Token (JWT), sigue los pasos a continuación. Esta solución asume que estás trabajando con un JWT firmado, que es el tipo más común, y se centra en proporcionar un enfoque claro y práctico para lograr tu objetivo.

Requisitos Previos

Primero, asegúrate de tener la biblioteca Nimbus JOSE + JWT agregada a tu proyecto. Si estás usando Maven, incluye la siguiente dependencia en tu `pom.xml`:

```
<dependency>
    <groupId>com.nimbusds</groupId>
    <artifactId>nimbus-jose-jwt</artifactId>
    <version>9.37</version> <!-- Reemplaza con la última versión -->
</dependency>
```

Puedes verificar la última versión en Maven Central.

Pasos para Extraer la Fecha de Expiración, la Fecha de Emisión y Otros Reclamos

Aquí te mostramos cómo analizar un JWT y recuperar la fecha de expiración, la fecha de emisión y reclamos adicionales usando la biblioteca `com.nimbusds.jwt`:

1. **Analizar la Cadena JWT**: Usa el método `SignedJWT.parse()` para convertir la cadena JWT en un objeto `SignedJWT`.
2. **Obtener el Conjunto de Reclamos**: Accede a los reclamos (pares clave-valor) del JWT usando `getJWTClaimsSet()`.
3. **Extraer Reclamos Específicos**:
   - Usa `getExpirationTime()` para la fecha de expiración (`exp` reclamo).
   - Usa `getIssueTime()` para la fecha de emisión (`iat` reclamo).
   - Usa `getSubject()`, `getClaim()` u otros métodos para reclamos adicionales.
4. **Manejar Errores**: Envuelve la lógica de análisis en un bloque try-catch para manejar posibles problemas de análisis.

Código de Ejemplo

A continuación, se muestra un ejemplo completo en Java que demuestra cómo extraer la fecha de expiración, la fecha de emisión y un reclamo adicional (por ejemplo, el sujeto) de un JWT:

```java
import com.nimbusds.jwt.SignedJWT;
import com.nimbusds.jwt.JWTClaimsSet;
import java.text.ParseException;
import java.util.Date;

public class JwtExtractor {

    public static void main(String[] args) {
        // Reemplaza esto con tu cadena JWT real
        String jwtString = "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwiaWF0IjoxNTE2MjM5N

        try {
            // Paso 1: Analizar la cadena JWT
            SignedJWT signedJWT = SignedJWT.parse(jwtString);

            // Paso 2: Obtener el conjunto de reclamos
            JWTClaimsSet claimsSet = signedJWT.getJWTClaimsSet();

            // Paso 3: Extraer fechas de expiración e emisión
            Date expirationDate = claimsSet.getExpirationTime();
            Date issuedDate = claimsSet.getIssueTime();
            String subject = claimsSet.getSubject(); // Ejemplo de otro reclamo

            // Paso 4: Mostrar los resultados
            if (expirationDate != null) {
                System.out.println("Fecha de expiración: " + expirationDate);
            } else {
                System.out.println("No se ha establecido fecha de expiración.");
            }

            if (issuedDate != null) {
                System.out.println("Fecha de emisión: " + issuedDate);
            } else {
                System.out.println("No se ha establecido fecha de emisión.");
            }
```

```
            if (subject != null) {
                System.out.println("Sujeto: " + subject);
            } else {
                System.out.println("No se ha establecido sujeto.");
            }

        } catch (ParseException e) {
            System.out.println("JWT no válido: " + e.getMessage());
        }
    }
}
```

Explicación del Código

1. Importaciones

- `SignedJWT`: Representa un JWT firmado y proporciona métodos para analizarlo y procesarlo.
- `JWTClaimsSet`: Contiene los reclamos del payload del JWT.
- `ParseException`: Lanzada si la cadena JWT está mal formada o no se puede analizar.
- `Date`: Usada para representar los tiempos de expiración e emisión.

2. Analizando el JWT

- El método `SignedJWT.parse(jwtString)` toma una cadena JWT (por ejemplo, `header.payload.signature`) y devuelve un objeto `SignedJWT`. Si el JWT es inválido, lanza una `ParseException`.

3. Accediendo a los Reclamos

- `signedJWT.getJWTClaimsSet()` recupera el conjunto de reclamos, que contiene todos los reclamos del payload del JWT.

4. Extraer Reclamos Específicos

- `getExpirationTime()`: Devuelve el reclamo `exp` como un objeto `Date` (o `null` si no está presente). Esto representa cuándo expira el token.
- `getIssueTime()`: Devuelve el reclamo `iat` como un objeto `Date` (o `null` si no está presente). Esto indica cuándo se emitió el token.

- `getSubject()`: Devuelve el reclamo `sub` como una cadena `String` (o `null` si no está presente), un ejemplo de otro reclamo estándar. También puedes usar `getClaim("key")` para recuperar reclamos personalizados como un `Object`.

5. Manejo de Errores

- El bloque try-catch asegura que si el JWT está mal formado o es inválido, el programa maneja el error de manera adecuada imprimiendo un mensaje de error.

Notas

- **JWT Firmados vs. No Firmados**: Este ejemplo usa `SignedJWT` para tokens firmados. Si tienes un JWT no firmado, usa `PlainJWT.parse(jwtString)` en su lugar. Para un enfoque más genérico, puedes usar `JWTParser.parse(jwtString)` y luego verificar el tipo (`SignedJWT`, `PlainJWT`, etc.) usando `instanceof`.
- **Verificación de Firma**: Este código no verifica la firma del JWT. En un entorno de producción, deberías verificar la firma usando `signedJWT.verify(verifier)` con una clave adecuada para asegurar la autenticidad del token.
- **Comprobaciones de Nulo**: Siempre verifica `null` al recuperar reclamos, ya que son opcionales y pueden no estar presentes en todos los JWT.
- **Reclamos Personalizados**: Para acceder a reclamos no estándar, usa `claimsSet.getClaim("claimName")`, que devuelve un `Object` que puedes convertir al tipo adecuado (por ejemplo, `String`, `Integer`).

Salida de Ejemplo

Para la cadena JWT de ejemplo anterior:

```
Fecha de expiración: mié. ene. 17 19:52:02 UTC 2018
Fecha de emisión: mié. ene. 17 19:50:22 UTC 2018
Sujeto: 1234567890
```

Este enfoque proporciona una manera simple y efectiva de extraer la fecha de expiración, la fecha de emisión y otros reclamos de un JWT usando la biblioteca `com.nimbusds.jwt`. ¡Reemplaza la `jwtString` con tu propio token y adapta el código a tus necesidades específicas!