快速入门

token的创建

创建SpringBoot工程，引入依赖

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.2.2.RELEASE</version>
        <relativePath/> <!-- lookup parent from repository -->
    </parent>
    <groupId>com.xxxx</groupId>
    <artifactId>jwtdemo</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <name>jwtdemo</name>
    <description>Demo project for Spring Boot</description>

    <properties>
        <java.version>1.8</java.version>
    </properties>

    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>
        <!--JWT依赖-->
        <dependency>
            <groupId>io.jsonwebtoken</groupId>
            <artifactId>jjwt</artifactId>
            <version>0.9.0</version>
        </dependency>

        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
            <exclusions>
                <exclusion>
                    <groupId>org.junit.vintage</groupId>
                    <artifactId>junit-vintage-engine</artifactId>
                </exclusion>
            </exclusions>
        </dependency>
```

```
   </dependencies>
</project>
```

创建测试类JwtTest，用于生成token：

```
package com.xxxx.jwtdemo;

import io.jsonwebtoken.JwtBuilder;
import io.jsonwebtoken.Jwts;
import io.jsonwebtoken.SignatureAlgorithm;
import io.jsonwebtoken.impl.Base64Codec;
import org.junit.jupiter.api.Test;
import org.springframework.boot.test.context.SpringBootTest;

import java.util.Date;

@SpringBootTest
public class JwtdemoApplicationTests {

   /**
    * 创建token
    */
   @Test
   public void testCreatToken() {
      //创建一个JwtBuilder对象
      JwtBuilder jwtBuilder = Jwts.builder()
            //声明的标识{"jti":"888"}
            .setId("888")
            //主体，用户{"sub":"Rose"}
            .setSubject("Rose")
            //创建日期{"ita":"xxxxxx"}
            .setIssuedAt(new Date())
            //签名手段，参数1：算法，参数2：盐
            .signWith(SignatureAlgorithm.HS256,"xxxx");
      //获取jwt的token
      String token = jwtBuilder.compact();
      System.out.println(token);

      //三部分的base64解密
      System.out.println("--------------------");
      String[] split = token.split("\\.");
      System.out.println(Base64Codec.BASE64.decodeToString(split[0]));
      System.out.println(Base64Codec.BASE64.decodeToString(split[1]));
      //无法解密
      System.out.println(Base64Codec.BASE64.decodeToString(split[2]));
   }
}
```

运行结果如下

```
eyJhbGciOiJIUzI1NiJ9.eyJqdGkiOiI4ODgiLCJzdWIiOiJSb3NlIiwiaWF0IjoxNTc4ODE0MjUyfQ.-FYFMHyfTcGzq900f_Drfdsges0ge2UjaWvPW9gCDto
--------------------
{"alg":"HS256"}
{"jti":"888","sub":"Rose","iat":1578814252
□�L□'�pl��M□□�ⱬ□���□�R6����� �
```

再次运行，会发现每次运行的结果是不一样的，因为我们的载荷中包含了时间