

一、发布 jwt 鉴权服务

1、选择 jwtToken

基本配置

* API名称

testapi

* API描述

test

安全认证

jwtTokenAuth

* 认证配置文件

```
{
  "apiTokenName": "apiToken",
  "userTokenName": "apiName",
  "secretKeys": [{"admin": "sinopec"}, {"test": "123456"}]
}
```

访问限制/单位

不限制

2、认证配置文件说明

需使用 JSON 对象描述

```
{
  "apiTokenName": "apiToken",          //非必须，请求头使用的 KEY，值为加密 token 串，默认 apiToken
  "userTokenName": "apiName",          //非必须，请求头使用的 KEY，值为用户名，默认 apiName
  "secretKeys": [{"admin": "secret"}, {"test": "123456"}], //必须，用户名及密钥数组
  "userTokenScope": "HEADER",          //非必须，用户请求 token 存放的位置，默认为 header
  "authFailContentType": "HTML_UTF8", //非必须，认证失败返回结果的 contentType，默认 json-utf8
  "authFailResult": "Unauthorized"     //非必须，认证失败返回结果
}
```

二、JAVA 程序生成 token 示例

Maven 工具类包

```
<dependency>
<groupId>com.auth0</groupId>
<artifactId>java-jwt</artifactId>
<version>3.1.0</version>
</dependency>
```

参考文章：

<https://blog.csdn.net/u011277123/article/details/78918390>

<https://yq.aliyun.com/php/1290>

```
JwtToken.java x
JwtToken createToken()
1 package com.ph.jwt;
2
3 import com.auth0.jwt.JWT;
4 import com.auth0.jwt.JWTVerifier;
5 import com.auth0.jwt.algorithms.Algorithm;
6 import com.auth0.jwt.interfaces.Claim;
7 import com.auth0.jwt.interfaces.DecodedJWT;
8
9 import java.util.Calendar;
10 import java.util.Date;
11 import java.util.HashMap;
12 import java.util.Map;
13
14 /**
15  * @author Free码农
16  * @version 1.0
17  * @since 2017-12-27
18  */
19 public class JwtToken {
20
21     /**
22      * 公用密钥-保存在服务端, 客户端是不会知道密钥的, 以防被攻击
23      */
24     public static String SECRET = "FreeMaNong";
25
26     /**
27      * 生成Token
28      *
29      * @return
30      * @throws Exception
31      */
32     public static String createToken() throws Exception{
33
34         // 签发时间
35         Date iatDate = new Date();
36
37         // 过期时间- 1分钟过期
38         Calendar nowTime = Calendar.getInstance();
39         nowTime.add(Calendar.MINUTE, 1);
40         Date expiresDate = nowTime.getTime();
41
42         Map<String, Object> map = new HashMap<String, Object>();
43         map.put("alg", "HS256");
44         map.put("typ", "JWT");
45         String token = JWT.create()
46             .withHeader(map)//header
47             .withClaim("name", "Free码农");//payload
48             .withClaim("age", "28")
49             .withClaim("org", "今日头条")
50             .withExpiresAt(expiresDate) // 设置过期时间-过期时间要大于签发时间
51             .withIssuedAt(iatDate) // 设置签发时间
52             .sign(Algorithm.HMAC256(SECRET));//加密
53         return token;
54     }
55
56     /**
57      * 解密Token
58      *
59      * @param token
60      * @throws Exception
61      */
62     public static Map<String, Claim> verifyToken(String token) throws Exception{
63         JWTVerifier verifier = JWT.require(Algorithm.HMAC256(SECRET))
64             .build();
65         DecodedJWT jwt = null;
66         try {
67             jwt = verifier.verify(token);
68         } catch (Exception e){
69             throw new RuntimeException("登录凭证已过去, 请重新登录");
70         }
71
72         return jwt.getClaims();
73     }
74 }
75
76
77
```

建议：出于安全考虑，请添加签发时间及过期时间

其它语言 SDK 工具包支持：（详情查看 <http://www.jwt.io>）

.NET
1C
C
C++
Clojure
Crystal
D
Delphi
Elixir
Erlang
Go
Groovy
Haskell
Haxe
Java
JavaScript
kdb+/Q
Lua
Node.js
Objective-C
Perl
PHP
PostgreSQL
Python
Ruby
Rust
Scala
Swift

三、工具测试方法

- 1、登录网址： <https://jwt.io/>
- 2、区域 2，3 不用更改
- 3、区域 4 写入密钥
- 4、区域 1 内容为 token 值

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.e30
.EQu-
qG0wFvi7NRflSUcUjQn01drSzwg7Xr1HerrNJ2Y

区域1

HEADER: ALGORITHM & TOKEN TYPE

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

区域2

PAYLOAD: DATA

```
{}
```

区域3

VERIFY SIGNATURE

```
HMACSHA256(
  base64UrlEncode(header) + "." +
  base64UrlEncode(payload),
  sinopec
)
```

区域4

☐ secret base64 encoded

SHARE JWT

```
{
  "status": 401,
  "msg": "Unauthorized",
  "data": null
}
```