传统的web应用中，我们通常通过cookie+session机制来保证调用的安全，在没有认证的情况下自动重定向到登录页面或者调用失败页面，而现在整个架构编程微服务模式了，cookie和session机制已经不能很好的满足保护API的需求了，更多的情况下采用token的验证机制，JWT的本质也是一种token。

JWT：JSON Web Token，是JSON风格的轻量级授权和认证规范，可以实现无状态，分布式的web应用授权。JWT的内容由三部分组成，分别是Header,Payload,Signature，三个部分之间通过.分割,举例

xxxxx.yyyyyy.zzzzz

Header

头部Header一般由2个部分组成alg和typ,alg是加密算法，如HMAC或SHA256,typ是token类型，取值为jwt,一个Header的例子

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

然后对Header部分进行Base64编码，得到第一部分的值

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.{PAYLOAD}.{SIGNATURE}

Payload

内容部分Payload是JWT存储信息的主体，包含三类内容

- 标准中注册的声明
- 公共的声明
- 私有的声明

标准中注册的声明

- iss:jwt签发者
- sub:jwt所面向的用户
- aud:接收jwt的一方
- exp:jwt的过期时间
- nbf:定义在什么时间之前该jwt是不可用的
- iat:jwt的签发时间
- jti:jwt的唯一标识，主要用作一次性token，避免重放攻击

公共的声明：

可以存放任何信息，根据业务实际需要添加，如用户id，名称等，但不要存放敏感信息

私有的声明：

私有声是提供者和消费者所共同定义的声明，不建议存放敏感信息

举例：定义一个payload:

{ "sub": "1234567890", "name": "John Doe", "admin": true }对其进行Base64编码，得到第二部分

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiYWRtaW4iOnRydWV9.{SIGNATURE}

Signature

token的签名部分由三部分组成256签名

```
var encodedString = base64UrlEncode(header) + '.' + base64UrlEncode(payload);
var signature = HMACSHA256(encodedString, 'secret');
```

得到最终的token串

eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzdWIiOiIxMjM0NTY3ODkwIiwibmFtZSI6IkpvaG4gRG9lIiwiYWRtaW4iOnRydWV9.TJVA95OrM7E2

1.Oauth2-Server中生成JWT Token

a)通过keytool生成证书

keytool -genkeypair -alias lizzy_key -keyalg RSA -keypass 123456 -keystore lizzy_key.jks -storepass 123456

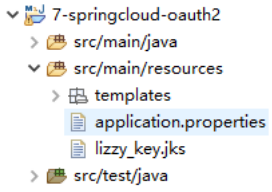
查看证书信息： keytool -list -v -keystore lizzy_key.jks -storepass 123456

```
C:\Users\cuilijian>keytool -list -v -keystore lizzy_key.jks -storepass 123456

密钥库类型: JKS
密钥库提供方: SUN

您的密钥库包含 1 个条目

别名: lizzy_key
创建日期: 2019-4-13
条目类型: PrivateKeyEntry
证书链长度: 1
证书[1]:
所有者: CN=崔立剑, OU=CU, O=CU, L=北京, ST=北京, C=CN
发布者: CN=崔立剑, OU=CU, O=CU, L=北京, ST=北京, C=CN
序列号: 56e3a0d
有效期开始日期: Sat Apr 13 15:55:07 CST 2019, 截止日期: Fri Jul 12 15:55:07 CST 2019
证书指纹:
         MD5: 12:57:E6:96:28:6E:A9:93:6A:A3:FC:6D:97:AD:DB:F5
         SHA1: 74:44:33:A4:94:68:77:8F:F4:39:74:16:D0:54:42:39:FC:80:9C:2B
         SHA256: 99:00:18:E4:1F:89:E7:0E:CB:3A:47:49:32:5A:6D:54:38:CF:2B:7A:1A:40:CC:7C:1B:62:AA:1A:3E:A0:E8:2C
         签名算法名称: SHA256withRSA
         版本: 3

扩展:

#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 9F 5E 4A 33 2D FC 4C 26   23 6D DA FC 7F 5E F2 B5  .^J3-.L&#m...^..
0010: 67 DD 81 27                                        g..'
]
]



*******************************************
*******************************************


C:\Users\cuilijian>
```

查看公钥信息：keytool -list -rfc -keystore lizzy_key.jks -storepass 123456

```
C:\Users\cuilijian>keytool -list -rfc -keystore lizzy_key.jks -storepass 123456

密钥库类型: JKS
密钥库提供方: SUN

您的密钥库包含 1 个条目

别名: lizzy_key
创建日期: 2019-4-13
条目类型: PrivateKeyEntry
证书链长度: 1
证书[1]:
-----BEGIN CERTIFICATE-----
MIIDWTCCAkGgAwIBAgIEBW46DTANBgkqhkiG9w0BAQsFADBdMQswCQYDVQQGEwJD
TjEPMA0GA1UECAwG5YyX5LqsMQ8wDQYDVQQHDAbljJfkuqwxCzAJBgNVBAoTAkNV
MQswCQYDVQQLEwJDVTESMBAGA1UEAwwJ5bSU56uL5YmRMB4XDTE5MDQxMzA3NTUw
NloXDTE5MDcxMjA3NTUwNlowXTELMAkGA1UEBhMCQO4xDzANBgNVBAgMBuWMl+S6
rDEPMA0GA1UEBwwG5YyX5LqsMQswCQYDVQQKEwJDVTELMAkGA1UECxMCQ1UxEjAQ
BgNVBAMMCeW0lOeri+WJkTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEB
AK8RJV2OaYh4bfx4unzHIyT2INmwy7JzIeq8/q84zKjE85DtluVp6SyOvnmyrQw9
ddcqoxXG8XMtqZFHQc1ErCpI+6zMc+m2g6hcPWElc5YP2MGJRlAld/TP2HVRqeXL
Ki6xFBxZVmVK4MU8b4cH+qbZ3ZmoFzFizn0/FUN3SMGlaMfEujlyF2xvxAldzpiK
e2Kic7dvkrMtcgho0rSWlx+9ZRYsfOhopZB/h17sS6CZYMVaf5udfwQ2bn1tmOMw
qVtYwO54gDzdFlw4xTVrbF68WXL7SxWAhU6OX6Aeq5rxFQEjavRCkTFPA9jzFLsV
RgoUDKGLhg3i9eiLz14PNj0CAwEAAaMhMB8wHQYDVR0OBBYEFJ9eSjMt/EwmI23a
/H9e8rVn3YEnMA0GCSqGSIb3DQEBCwUAA4IBAQAP2jRUthAHqMkgTREoPq5tH30v
kCmhPCRiZiJ1mOYA54HIGOBVlebjzlbKvrxmmhtaofgtV/zfqxDjjlCDN3Xw+z8I
llhGJoVYg+xSoS9H8sI1Pr/guH16E2UhpyuvJ8Fw2fCfdhFcOAC4I/29OycfxAdx
a7Lyg3XBpwdXvOkyQug1b3TpeMlKOYiUsFMi2W4dUuLRb4mna7kb8TtHiRSXTllA
ksf+puyLliLSKAmgOs95Rk9N6CYAW9P1dKJ1ECGILENUDIDgl65z+J3T/4NWK1+s
i0jJxB8KHn1zRHuTsEnqhIdYoGAVYCDPAyYuXPOSogiPrG+NEFNbdo+OK+QC
-----END CERTIFICATE-----


*******************************************
*******************************************
```

b)将生成的kevin_key.jks文件放到oauth2-server工程的srce/main/resources目录下



c)添加jwt相关jar包依赖

```
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-jwt</artifactId>
</dependency>
```

d)在OAuth2服务器端配置核心类AuthorizationServerConfiguration中增加jwt token相关配置

```
@Override
        public void configure(final AuthorizationServerEndpointsConfigurer endpoints) throws Exception {
                // @formatter:off
                endpoints.authenticationManager(authenticationManager)
                        .userDetailsService(userDetailsService)
                        //加入JWT
                        .accessTokenConverter(accessTokenConverter());
        }
        /**
         * jwt配置
         * @return
         */
        @Bean
        public JwtAccessTokenConverter accessTokenConverter() {
                JwtAccessTokenConverter converter = new JwtAccessTokenConverter() {
                        @Override
                        public OAuth2AccessToken enhance(OAuth2AccessToken accessToken, OAuth2Authentication
authentication) {
                                String username = authentication.getName();
                                final Map<String, Object> additionalInformation  = new HashMap<>();
                                additionalInformation.put("user_name", username);
                                ((DefaultOAuth2AccessToken) accessToken).setAdditionalInformation(additionalInformation);
                                OAuth2AccessToken token = super.enhance(accessToken, authentication);
                                return token;
                        }
                };
                KeyPair keyPair = new KeyStoreKeyFactory(new ClassPathResource("lizzy_key.jks"),
"123456".toCharArray()).getKeyPair("lizzy_key");
                converter.setKeyPair(keyPair);
                return converter;
        }
```

2.测试Oauth2服务

http://localhost:8888/oauth/authorize?response_type=code&client_id=client&redirect_uri=http://baidu.com&state=123

出现登录页面，输入用户名：admin 密码：123456

点击Submit按钮，进入用户授权确认页面

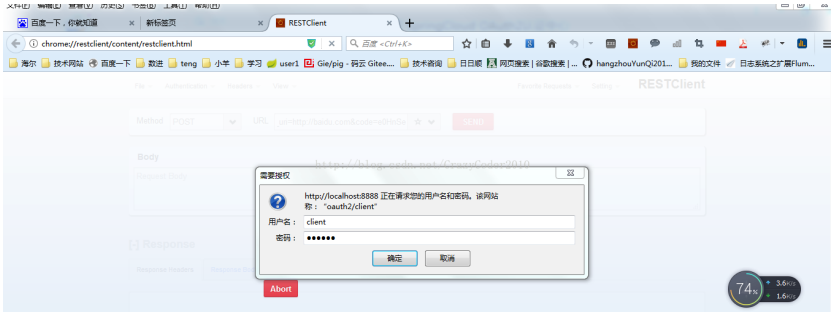
点击Approve,跳转到baidu页面，后面携带了code和state参数

https://www.baidu.com/?code=F7LsMB&state=123

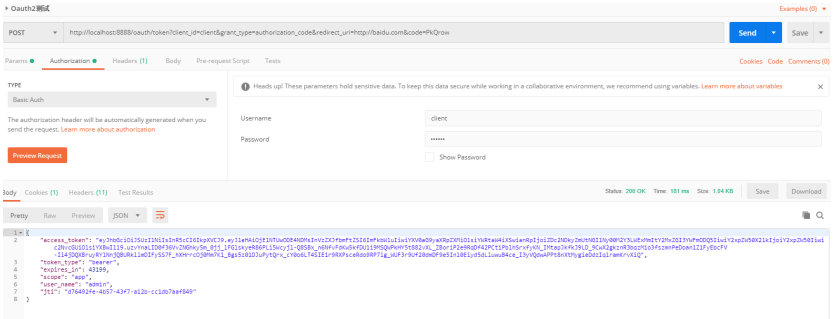
根据code换取access_code，注意使用post方法

http://localhost:8888/oauth/token?client_id=client&grant_type=authorization_code&redirect_uri=http://baidu.com&code=F7LsMB

注意这个code要和上个步骤中获得的code保持一致



用户名输入client，密码是secret,点击确定,可以看到access_token已经是jwt格式的字符了



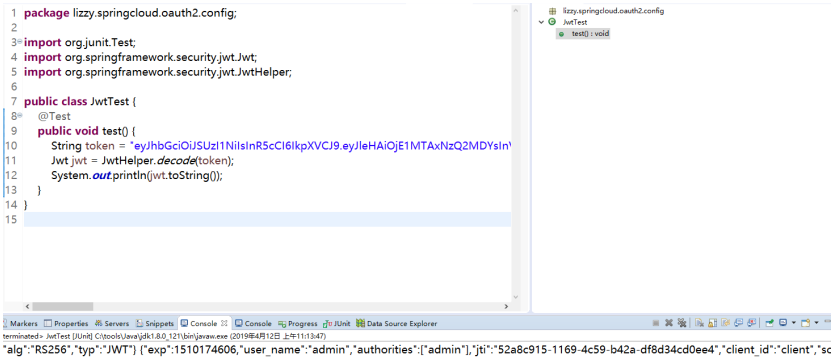
{ "access_token":
"eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE1NTUwODE4NDMsInVzZXJfbmFtZSI6ImFkbWluIiwiYXV0aG9yaXRpZXMiOlsiYWRtaW4iXSwi
Q8SBx_n6NfvFdKw5kfDU1i9MSQWPkHY5t882vXL_ZBoriP2e9RqDf42PCtiPblhSrxfyKN_IMtapJkfkJ9LD_9CwX2gkznR3bqzMio3fszmnPeDoan1Z1FyEbc
Ii4jDQXBruyRYlNnjQBURkllmDIFySS7F_hXHrrcOjOMm7K1_Bgs5z01DJuPytQrx_cY0o6LT4SIE1r9RXPsceRdo9RP71g_WUF3r9Uf20dmDF9e5Inl0Eiyd5
"token_type": "bearer",
"expires_in": 43199,
"scope": "app",
"user_name": "admin",
"jti": "d76492fe-4b57-43f7-a12b-cc1db7aaf849"
}

3.access_token信息解析

我们通过上个步骤得到的token信息是不可读的，但是因为header,body都是经过base64转码过的，因此我们可以通过Base64将其解码，

spring cloud里也提供了jwt相关的工具类帮我们来反解析这个串

```
package lizzy.springcloud.oauth2.config;
import org.junit.Test;
import org.springframework.security.jwt.Jwt;
import org.springframework.security.jwt.JwtHelper;
public class JwtTest {
        @Test
        public void test() {
                String token =
"eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE1MTAxNzQ2MDYsInVzZXJfbmFtZSI6ImFkbWluIiwiYXV0aG9yaXRpZXMiOlsiYWRtaW4iXSwi
FHApwznyYl-D3IjBOTpjhdhUXfYv782kfS9vdTOVZsu2HN-MGb-N-6Hf0efZ_mmz54IahJaq3KTw251v4L2O5A1r_iMuUP7GXs_qPHAGn3K1b4l-
mNnpJdH5hhS5zYIRqOX2a8DXyI4zD7g8BQL-9PiR3kj9k_z9nW8vY912_x5Kyoc-
sehxxQ5uQHM3xu6DzOwBpbbER7U_NnUwmcz5nS9YyAexSDnBbZAVpQavL2s1yYQVMJ5Dreq2asXHFbeQHXu5UqVbbTFuOgAylbFJ9K-
3nsGAKT9NbzqBPRovI3s_X9HgjrzJHAuojBMeKOQMbvYSbUg2HB7MNNJw";
                Jwt jwt = JwtHelper.decode(token);
                System.out.println(jwt.toString());
        }
}
```



通过这个测试我们可以看出来，token中已经包含了当前用户的信息了，包括我们在accessTokenConvertor()方法中给token中添加的额外信息user_name