JEECG平台对外接口应用文档

JEECG
2018-01-03

JEECG 开发平台 API

一、接口方式

接口调用采用 http 协议，rest 请求方式；

二、接口安全

接口安全采用 Json web token (JWT)机制，基于 token 的鉴权机制.

1. 机制说明

基于 token 的鉴权机制类似于 http 协议也是无状态的，它不需要在服务端去保留用户的认证信息或者会话信息。这就意味着基于 token 认证机制的应用不需要去考虑用户在哪一台服务器登录了，这就为应用的扩展提供了便利。

2. 基本流程

流程上是这样的：

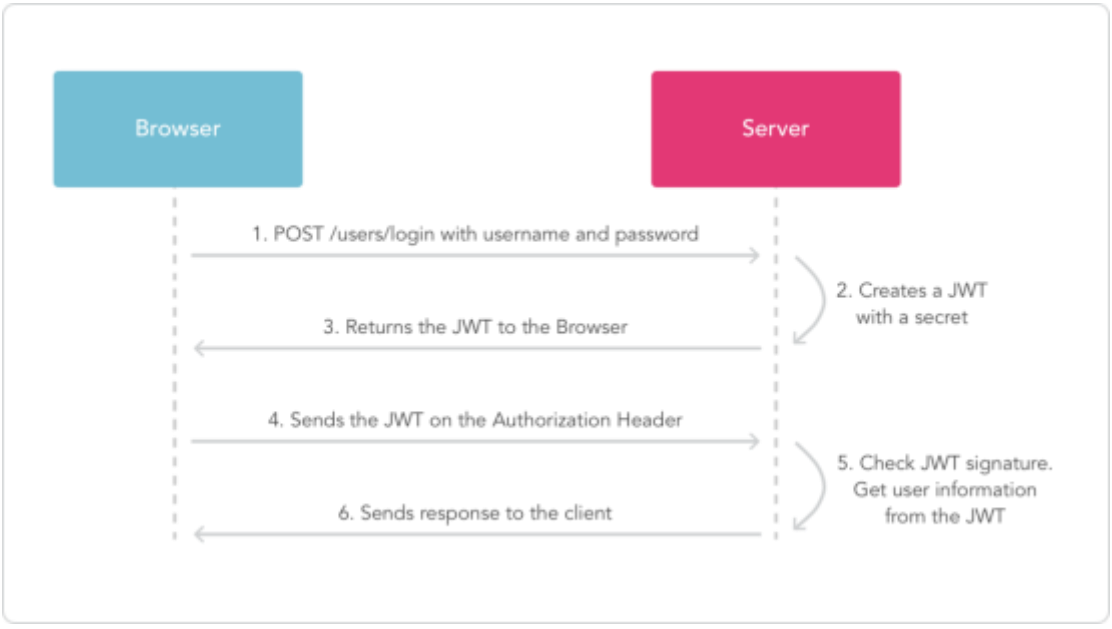
（1） 用户使用用户名密码来请求服务器

（2） 服务器进行验证用户的信息

（3） 服务器通过验证发送给用户一个 token

（4） 客户端存储 token ,并在每次请求时附送上这个 token 值(存在 head 里的参数 X-AUTH-TOKEN)

（5） 服务端验证 token 值，并返回数据



3. 优点

- 因为 json 的通用性，所以 JWT 是可以进行跨语言支持的，像 JAVA,JavaScript,NodeJS,PHP 等很多语言都可以使用。
- 因为有了 payload 部分，所以 JWT 可以在自身存储一些其他业务逻辑所必要的非敏感信息。
- 便于传输，jwt 的构成非常简单，字节占用很小，所以它是非常便于传输的。
- 它不需要在服务端保存会话信息，所以它易于应用的扩展

4. **安全相关**

- 不应该在 jwt 的 payload 部分存放敏感信息，因为该部分是客户端可解密的部分。
- 保护好 secret 私钥，该私钥非常重要。
- 如果可以，请使用 https 协议

三、 缓存配置

JWT 验证 token 采用 redis 进行缓存，

redis 配置文件：src/main/resources/redis.properties

修改 redis 对应的 IP 和端口。

```
#redis
redis.host=124.206.91.99
redis.port=6379
redis.pass=
redis.adapter.maxIdle=100
redis.adapter.minIdle=10
redis.adapter.testOnBorrow=true
redis.adapter.testOnReturn=true
redis.adapter.testWhileIdle=true
redis.adapter.numTestsPerEvictionRun=10
redis.adapter.timeBetweenEvictionRunsMillis=60000
```

四、 接口说明

注意：访问除【鉴权 TOKEN 接口】以外的接口时，都需要访问用户拥有对接口的访问权限，如无权限，将直接返回如下信息：

{"message":"您没有该接口的权限！","data":null,"ok":false,"respCode":"-1"}

1. 鉴权 TOKEN 接口

■描述

根据用户名和密码获取 TOKEN。

■访问地址

`http://域名/rest/tokens`

■访问方式

GET

■参数

参数名	数据类型	是否必须	示例值	默认值	描述
username	String	Y	“admin”		用户名
password	String	Y	“123456”		密码

■返回值

成功时，直接返回 token 字符串。

失败时，直接返回用户账号密码错误!

■校验规则

无

■请求示例

请求地址：`http://域名/rest/tokens`

```
{
    "username":"admin",
    "password":"123456"
}
```

■返回示例

```
成功案例：
eyJhbGciOiJIUzI1NiJ9.eyJqdGkiOiI4YThhYjBiMjQ2ZGM4MTEyMDE0NmRjODE4MTk1MDA1MiIsInN1YiI6ImFkbWluIiwiaWF0IjoxNTExODU0NDE4fQ.tnILZEivS-6YOX9uqsnCHygh7-XrG_-Sj8vLslNGkdQ
失败案例：
用户账号密码错误!
```

2. 创建黑名单信息接口

■描述

创建黑名单信息接口，黑名单为单表。

■访问地址

`http://域名/rest/tsBlackListController`

■访问方式

POST

■参数（详见 excel）

参数名	数据类型	是否必须	示例值	默认值	描述
ip	String	Y	“192.168.1.1”		
……			……省略信息其他字段……		

■返回值

参数名	描述
respCode	返回码（见附录 1 接口返回信息列表）
respMsg	返回信息（见附录 1 接口返回信息列表）
data	返回结果（NULL）
ok	状态

■校验规则

1. 接口中涉及日期时间的字段，要求格式化为字符串传递，日期格式为“YYYY-MM-dd”，时间格式为“YYYY-MM-dd HH:mm:ss”。

■请求示例

请求地址：`http://域名/rest/tsBlackListController`

参数如下：

注意：创建企业无需传 id，子表无需传 id 和企业 id，这些都会在后台生成，必需要传入的是来源 id 和来源表。

```
{
   "ip": "192.1.1.1",
……(省略信息其他字段)
}
```

■返回示例

成功案例：

```
{
    "respCode":"0",
    " respMsg":"成功"
}
```

失败案例：

```
{
    "respCode":"-1",
    "respMsg":"黑名单创建失败"
}
```

3. 查询黑名单信息接口

■描述

根据 id 查询或查询黑名单信息接口。

■访问地址

根据 id 查询	`http://域名/rest/tsBlackListController/get/{id}`

■访问方式

GET

■参数

无

■返回值

参数名	描述
respCode	返回码（见附录 1 接口返回信息列表）
respMsg	返回信息（见附录 1 接口返回信息列表）
data	返回结果（结构参照创建企业接口的参数，具体字段参照 excel）
ok	状态

■校验规则

■请求示例

请求地址：`http://域名/rest/tsBlackListController/get/297e7ae15f7f7f7e015f7fb0f57e0040`

■返回示例

成功案例:


```
{
    "message": "成功",
    "data": {
        "id": "402881f15e751d2a015e75212c570005",
        "createBy": "admin",
        "updateBy": "",
        "bpmStatus": "1",
        "ip": "111.193.210.4",
        "createName": "管理员",
        "createDate": "2017-09-12 16:07:41",
        "updateName": "",
        "updateDate": null,
        "sysOrgCode": "A03",
        "sysCompanyCode": "A03"
    },
    "respCode": "0",
    "ok": true
}
```


失败案例：


```
{"data":null,"respCode":"-1","respMsg":"根据所传 id 查询无结果"}
```


4. 修改黑名单信息接口

■描述

根据 id 修改

■访问地址

`http://域名/rest/tsBlackListController/update/{id}`

■访问方式

PUT

■参数

参数名	数据类型	是否必须	示例值	默认值	描述
id	String	Y	“402881f15f811877015f8124ca1c0002”		
ip	String	Y	“192.168.1.1”		

参数名	数据类型	是否必须	示例值	默认值	描述
			……省略信息其他字段……		

■返回值

参数名	描述
respCode	返回码（见附录 1 接口返回信息列表）
respMsg	返回信息（见附录 1 接口返回信息列表）
data	返回结果（NULL）
ok	状态

■校验规则

1.通过校验主表的字段：来源 id 和来源表验证数据唯一性。

■请求示例

请求地址：`http://域名/rest/tsBlackListController/update/402881f15f811877015f8124ca1c0002`

参数如下：

```
{
   "id": "402881e75f94878e015f94896bb80002",
   "ip": "1.1.1.1"

}
```

■返回示例

```
成功案例：
{
     "respCode":"0",
     "respMsg":"成功"
}

失败案例：
{
     "respCode":"-1",
     "respMsg":"输入 ID 无效,重复输入"
}
```

5. 删除黑名单接口

■描述

根据 id 删除

■访问地址

`http://域名/rest/tsBlackListController/delete/{id}`

■访问方式

DELETE

■参数

无

■返回值

参数名	描述
respCode	返回码（见附录 1 接口返回信息列表）
respMsg	返回信息（见附录 1 接口返回信息列表）
data	返回结果（NULL）
ok	状态

■校验规则

无

■请求示例

请求地址：`http://域名/rest/tsBlackListController/delete/297e7ae15f7f7f7e015f7fb0f57e0040`

■返回示例

```
成功案例：
{
    "respCode":"0",
    "respMsg":"成功"
}
失败案例：
{
    "respCode":"-1",
    "respMsg":"输入 ID 无效,重复输入"
}
```

五、客户端测试代码

代码示例

```
public static String getToken(String userName,String password){
          String url =
"http://localhost:8080/jeecg/rest/tokens?username="+userName+"&password="+password;
          String token= JwtHttpUtil.httpRequest(url, "POST", null);
          return token;
     }
          //获取黑名单列表
     public static JSONObject getBlackList(String token){
          String url = "http://localhost:8080/jeecg/rest/tsBlackListController";
          JSONObject resp= JwtHttpUtil.httpRequest(url, "GET", null,token);
          return resp;
     }
     //创建黑名单
     public static JSONObject createBlackList(String token,String json){
          String url = "http://localhost:8080/jeecg/rest/tsBlackListController";
          JSONObject resp= JwtHttpUtil.httpRequest(url, "POST", json,token);
          return resp;
     }
     //更新黑名单
     public static JSONObject updateBlackList(String token,String json){
          String url = "http://localhost:8080/jeecg/rest/tsBlackListController";
          JSONObject resp= JwtHttpUtil.httpRequest(url, "PUT", json,token);
          return resp;
     }
     //删除黑名单
     public static JSONObject deleteBlackList(String token,String id){
          String url = "http://localhost:8080/jeecg/rest/tsBlackListController/"+id;
          JSONObject resp= JwtHttpUtil.httpRequest(url, "DELETE", null,token);
          return resp;
     }
     //查询黑名单
     public static JSONObject getBlackList(String token,String id){
          String url = "http://localhost:8080/jeecg/rest/tsBlackListController/"+id;
          JSONObject resp= JwtHttpUtil.httpRequest(url, "GET", null,token);
          return resp;
     }
```

参考源码：


JwtHttpUtil.java


JwtRestfulClientDemo.java

附录 1：

接口返回 CODE

code	msg	说明	解决方案
0	SUCCESS	成功	
-1	ERROR	无接口访问权限	
1000	VALID_ERROR	验证失败	
r0001	SAVE_SUCCESS	写入成功	
r0002	UPDATE_SUCCESS	更新成功	
r0003	REMOVE_SUCCESS	删除成功	