敏捷RABC作业

数据库设计

```sql
-- 创建User表
CREATE TABLE User (
  Id INT PRIMARY KEY AUTO_INCREMENT,
  Username VARCHAR(255),
  Password VARCHAR(255)
);

-- 创建Role表
CREATE TABLE Role (
  Id INT PRIMARY KEY AUTO_INCREMENT,
  role_name VARCHAR(255)
);

-- 创建UserRole表
CREATE TABLE UserRole (
  Id INT PRIMARY KEY AUTO_INCREMENT,
  user_id INT,
  role_id INT
);
```

```sql

-- 创建权限表
CREATE TABLE Permission (
  Id INT PRIMARY KEY AUTO_INCREMENT,
  permission_name VARCHAR(255)
);

-- 创建RolePermission表
CREATE TABLE RolePermission (
  Id INT PRIMARY KEY AUTO_INCREMENT,
  role_id INT,
  permission_id INT
);

```

数据库操作

Usermapper.xml文件如下

```xml
<select id="findRole" resultType="String">

    SELECT r.role_name
    FROM User u
             JOIN UserRole ur ON u.Id = ur.user_id
             JOIN Role r ON r.Id = ur.role_id
    WHERE u.Username = #{username};
        </select>


    <select id="findPermission" resultType="String">

        SELECT p.permission_name
        FROM User u
                 JOIN UserRole ur ON u.Id = ur.user_id
```

```
                    JOIN RolePermission rp ON rp.role_id = ur.role_id
                    JOIN Permission p ON p.Id = rp.permission_id
          WHERE u.Username = #{username};

    </select>
```

Usermapper.java如下

```java
String findRole(String username);
List<String> findPermission(String username);
```

控制器

```java
package com.web.test.Controller;


import com.web.test.Mapper.UserMapper;
import com.web.test.Service.RedisService;
import com.web.test.VO.CommonResult;
import com.web.test.VO.ResultVO;
import com.web.test.VO.SendVO;
import io.swagger.annotations.Api;
import io.swagger.annotations.ApiOperation;
import org.slf4j.Logger;
import org.slf4j.LoggerFactory;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

```

```java
@Api(tags = "演示接口")
@RestController
@RequestMapping("/my-api/demo")
public class DemoServer {
    RedisService redisService;

    @Autowired
    private UserMapper userMapper;
    private final Logger logger = LoggerFactory.getLogger(getClass());
    @ApiOperation("Hello World接口")
    @GetMapping("/hello")
    public CommonResult<?> hello(@RequestBody SendVO sendVO) {
        ResultVO resultVO=new ResultVO();
        logger.info(sendVO.getUsername());
        if (sendVO != null && sendVO.getUsername() != null) {
            resultVO.setRole(userMapper.findRole(sendVO.getUsername()));
            resultVO.setPermissions(userMapper.findPermission(sendVO.getUsername()));
        } else {
            // handle the case where sendVO or sendVO.getUsername() is null
        }


        return CommonResult.success(resultVO);
    }


    @GetMapping("/change")
    public CommonResult<?> change(){
        return CommonResult.success("OK");
    }
}
```

相关类 `SendVO`

```java
package com.web.test.VO;


import lombok.Data;

@Data
public class SendVO {

    private String Username;

    private String password;


    public SendVO(){

    }


}

```

`ResultVO`

```java
package com.web.test.VO;

import lombok.Data;

import java.util.List;

```

```java
@Data
public class ResultVO {

    private String role;

    private List<String> permissions;

}

```

拦截器

`MyInterceptor`

```java
package com.web.test;

import com.web.test.Mapper.UserMapper;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Component;
import org.springframework.web.servlet.HandlerInterceptor;
import org.springframework.web.servlet.ModelAndView;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@Component
public class MyInterceptor implements HandlerInterceptor {


    private UserMapper userMapper;

    @Autowired
    public MyInterceptor(UserMapper userMapper) {
        this.userMapper = userMapper;
    }
    @Override
    public boolean preHandle(HttpServletRequest request, HttpServletResponse response, Object handler)
            throws Exception {
        // 在请求处理之前执行的操作
        System.out.println("=====preHandle=====");


        String username=request.getHeader( "Authorization");
        String uri = request.getRequestURI();
        System.out.println(uri);
        System.out.println(username);
        String role=userMapper.findRole(username);
        if ("/my-api/demo/change".equals(uri) && !"老师".equals(role)) {
            // 如果用户试图访问/change URI，并且他们的角色不是Teacher，那么终止请求处理
            response.setStatus(HttpServletResponse.SC_FORBIDDEN);
            response.getWriter().write("You do not have the necessary permissions to access this resource.");
            return false;
        } else {
            // 对于所有其他的请求，继续处理
            return true;
        }

    }

    @Override
    public void postHandle(HttpServletRequest request, HttpServletResponse response, Object handler,
                           ModelAndView modelAndView) throws Exception {
        // 在请求处理之后，视图渲染之前执行的操作
        System.out.println("=====postHandle=====");
    }

    @Override
    public void afterCompletion(HttpServletRequest request, HttpServletResponse response, Object handler,
                                Exception ex) throws Exception {
        // 在请求完成之后执行的操作，即视图渲染完成后
        System.out.println("=====afterCompletion=====");
    }
}


```

`MyInterceptorConfig` 类

```java
package com.web.test.Config;

import com.web.test.Mapper.UserMapper;
import com.web.test.MyInterceptor;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Configuration;
import org.springframework.web.servlet.config.annotation.InterceptorRegistry;
import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;

@Configuration
public class MyInterceptorConfig implements WebMvcConfigurer {

    @Autowired
    private UserMapper userMapper;
    @Override
    public void addInterceptors(InterceptorRegistry registry) {
        registry.addInterceptor(new MyInterceptor(userMapper))
```

```
                    .addPathPatterns("/my-api/**"); // 设置拦截的URL模式
    }
}

```

演示

role设置成两种

user=ooo role=老师

user=hhh role=学生

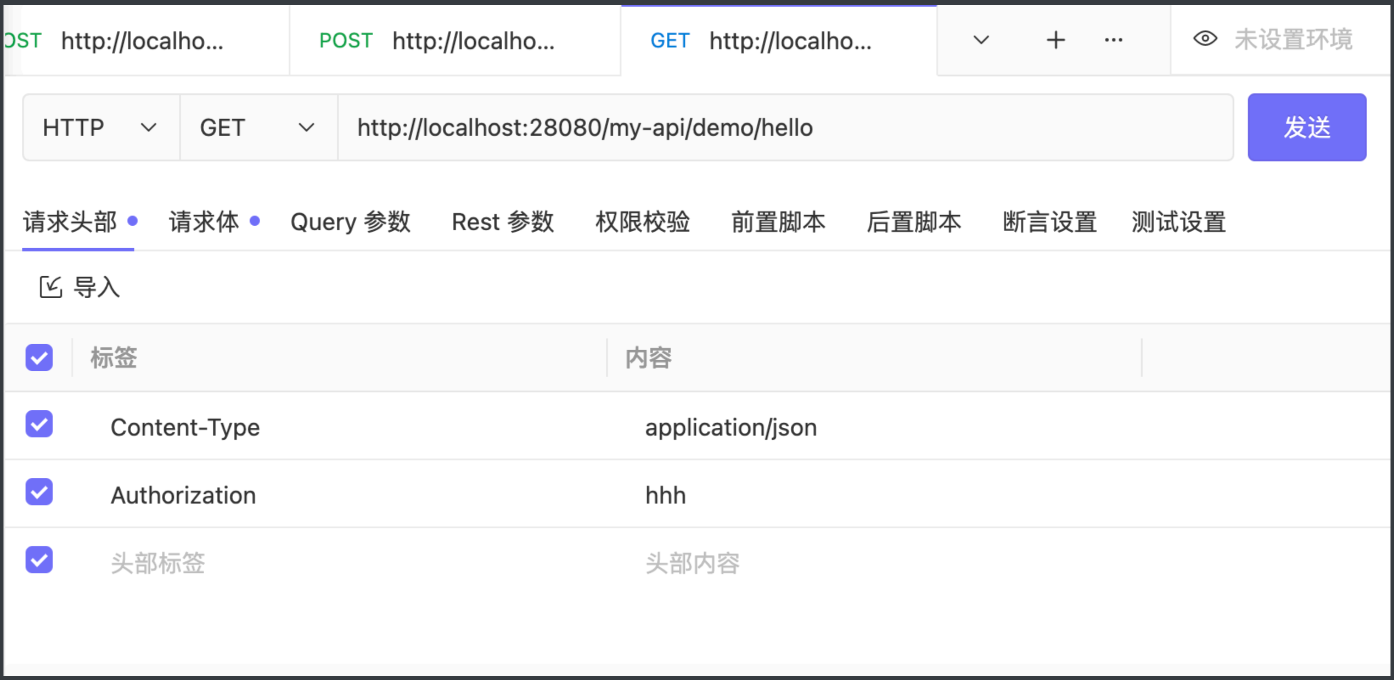
role=老师

都可以访问

注意请求头部也要加上(拦截器用作username)

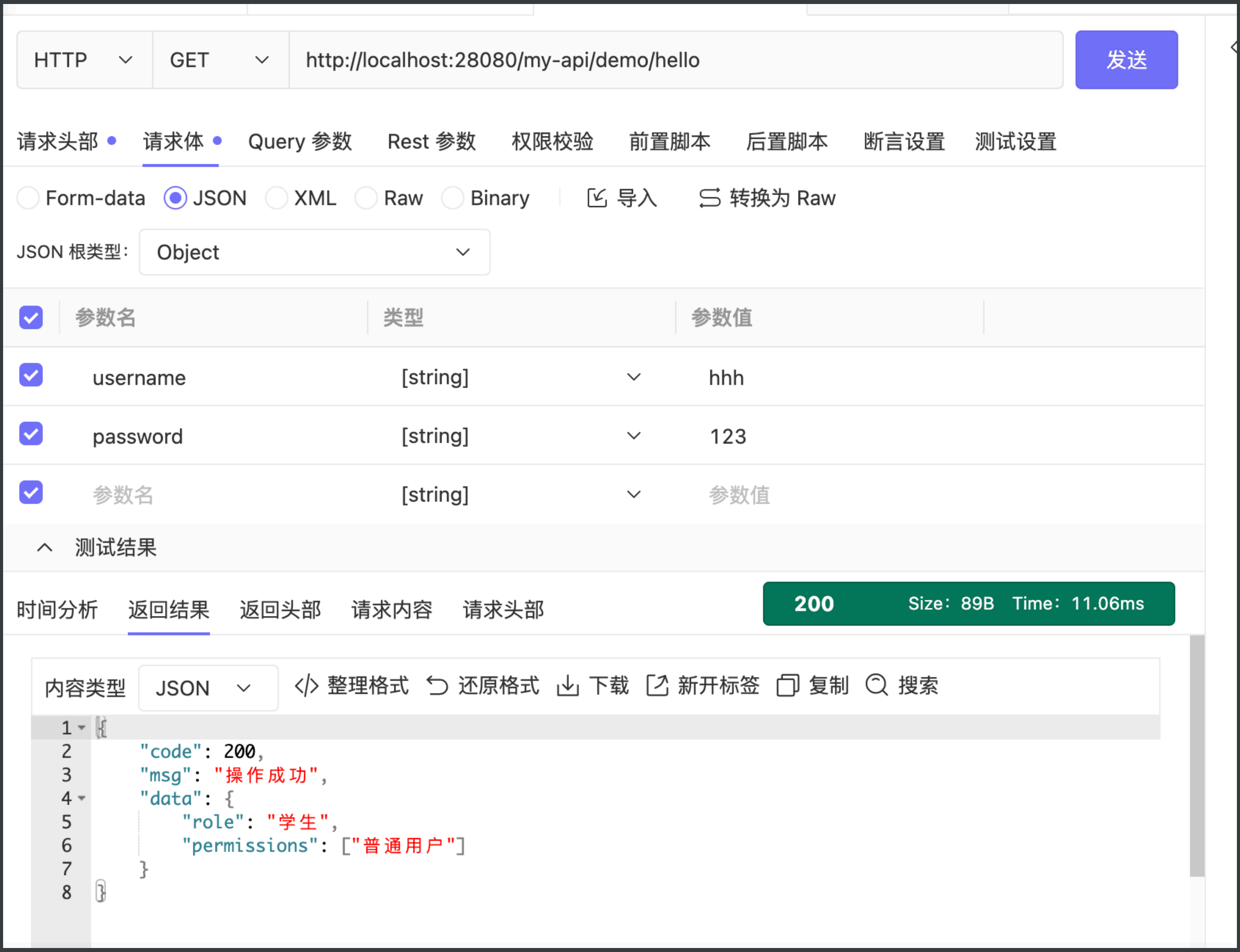
role=学生

change不可以访问



GET http://localhost:28080/my-api/demo/hello

标签	内容
✓ Content-Type	application/json
✓ Authorization	hhh
✓ 头部标签	头部内容



GET http://localhost:28080/my-api/demo/hello

Form-data [✓ JSON] XML Raw Binary

JSON 根类型: Object

参数名	类型	参数值
✓ username	[string]	hhh
✓ password	[string]	123
✓ 参数名	[string]	参数值

200 Size: 89B Time: 11.06ms

内容类型 JSON

```
{
    "code": 200,
    "msg": "操作成功",
    "data": {
        "role": "学生",
        "permissions": ["普通用户"]
    }
}
```