

8-4 保存角色数据（后端）

140.143.132.225:8000/project-1/doc-142

一、编写持久层代码

因为在 `role.vue` 页面保存角色数据，分为两种情况：保存新建角色，修改已有角色。分别对应MySQL的INSERT和UPDATE操作。所以我们要把两种SQL语句都定义出来。

编写 `TbRole.xml` 文件，定义SQL语句

```
1. <insert id="insertRole" parameterType="TbRole">
2.     INSERT INTO tb_role
3.         SET role_name=#{roleName}, permissions=#{permissions}
4. </insert>
5. <update id="updateRolePermissions" parameterType="TbRole">
6.     UPDATE tb_role
7.         SET permissions=#{permissions}
8.         WHERE id=#{id}
9. </update>
```

编写 `TbRoleDao.java` 接口中的抽象方法

```
1. public interface TbRoleDao {
2.     .....
3.     public int insertRole(TbRole role);
4.     public int updateRolePermissions(TbRole role);
5. }
```

二、编写业务层代码

在 `RoleService.java` 接口中定义抽象方法

```
1. public interface RoleService {
2.     .....
3.     public void insertRole(TbRole role);
4.     public void updateRolePermissions(TbRole role);
5. }
```


在 `RoleServiceImpl.java` 类中实现抽象方法

```
1. public class RoleServiceImpl implements RoleService {
2.     .....
3.     @Override
4.     public void insertRole(TbRole role) {
5.         int row = roleDao.insertRole(role);
6.         if (row != 1) {
7.             throw new EmosException("添加角色失败");
8.         }
9.     }
10.    @Override
11.    public void updateRolePermissions(TbRole role) {
12.        int row = roleDao.updateRolePermissions(role);
13.        if (row != 1) {
14.            throw new EmosException("修改角色失败");
15.        }
16.    }
17. }
```

三、编写Web层代码

创建 `InsertRoleForm.java` 类，接收移动端数据

```
1. @Data
2. @ApiModel
3. public class InsertRoleForm {
4.     @NotBlank
5.     private String roleName;
6.     @NotBlank
7.     private String permissions;
8. }
```



创建 `UpdateRolePermissionsForm.java` 类，接收移动端数据

```
1. @Data
2. @ApiModel
3. public class UpdateRolePermissionsForm {
4.     @NotNull
5.     @Min(1)
6.     private Integer id;
7.     @NotBlank
8.     private String permissions;
9. }
```

在 `RoleController.java` 中声明Web方法

```
1. public class RoleController {
2.     .....
3.     @PostMapping("/insertRole")
4.     @ApiOperation("添加角色")
5.     @RequiresPermissions(value = {"ROOT","ROLE:INSERT"},logical = Logical.OR)
6.     public R insertRole(@Valid @RequestBody InsertRoleForm form) {
7.         if(!JSONUtil.isJsonArray(form.getPermissions())){
8.             throw new EmosException("权限不是数组格式");
9.         }
10.        TbRole entity=new TbRole();
11.        entity.setRoleName(form.getRoleName());
12.        entity.setPermissions(form.getPermissions());
13.        roleService.insertRole(entity);
14.        return R.ok().put("result","success");
15.    }
16.    @PostMapping("/updateRolePermissions")
17.    @ApiOperation("修改角色")
18.    @RequiresPermissions(value = {"ROOT","ROLE:UPDATE"},logical = Logical.OR)
19.    public R updateRolePermissions(@Valid @RequestBody UpdateRolePermissionsForm form) {
20.        if(!JSONUtil.isJsonArray(form.getPermissions())){
21.            throw new EmosException("权限不是数组格式");
22.        }
23.        TbRole entity=new TbRole();
24.        entity.setId(form.getId());
25.        entity.setPermissions(form.getPermissions());
26.        roleService.updateRolePermissions(entity);
27.        return R.ok().put("result","success");
28.    }
29. }
```

