第二章 Shiro权限管理

主讲：泽林.王峰（2019.4.13）

授课大纲

1、SpringBoot简介

2、在Idea中搭建SpringBoot工程

3、在pom.xml文件中引入相关依赖（mybatis、通过Mapper、分页插件）

4、在application.yml中完成SpringBoot相关配置

5、如何切换数据源

6、完成上章的用户认证功能

授课内容

1、SpringBoot简介

```
    Spring Boot是由Pivotal团队提供的全新框架，其设计目的是用来简化新Spring应用的初始搭建以及开发过程。该框架使用了特定的方式来进行配置，从而使开发人员不再需要定义样板化的配置。通过这种方式，Spring Boot致力于在蓬勃发展的快速应用开发领域(rapid application development)成为领导者。
```

2、在Idea中搭建SpringBoot工程

2.1）以maven创建普通java工程（在此略过）。

2.2）配置pom.xml文件

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>com.zelin</groupId>
    <artifactId>Shiro_02</artifactId>
    <version>1.0-SNAPSHOT</version>
     <!--1.配置下面的依赖所用到的父工程-->
    <parent>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-parent</artifactId>
        <version>2.0.1.RELEASE</version>
```

```xml
            <relativePath/> <!-- lookup parent from repository -->
        </parent>
        <properties>
            <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
            <java.version>1.8</java.version>
        </properties>
        <dependencies>
            <dependency>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-starter</artifactId>
            </dependency>
            <!--2.与junit进行整合的starter-->
            <dependency>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-starter-test</artifactId>
                <scope>test</scope>
            </dependency>
            <!--3.代表springboot与web的整合的starter-->
            <dependency>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-starter-web</artifactId>
            </dependency>

            <!--4.添加数据库-->
            <dependency>
                <groupId>mysql</groupId>
                <artifactId>mysql-connector-java</artifactId>
                <version>5.1.47</version>
            </dependency>
            <!--5.添加mybatis与springboot整合的依赖-->
            <dependency>
                <groupId>org.mybatis.spring.boot</groupId>
                <artifactId>mybatis-spring-boot-starter</artifactId>
                <version>2.0.1</version>
            </dependency>
            <!-- 6.添加mybatis分页插件与springboot整合的依赖 -->
            <dependency>
                <groupId>com.github.pagehelper</groupId>
                <artifactId>pagehelper-spring-boot-starter</artifactId>
                <version>1.2.10</version>
            </dependency>
            <!-- 7.添加通过mapper与springboot整合的依赖 -->
            <dependency>
                <groupId>tk.mybatis</groupId>
                <artifactId>mapper-spring-boot-starter</artifactId>
                <version>2.1.5</version>
            </dependency>

            <dependency>
                <groupId>org.apache.commons</groupId>
                <artifactId>commons-lang3</artifactId>
                <version>3.3.2</version>
            </dependency>
```

```xml
    </dependencies>
    <build>
        <plugins>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
            </plugin>
        </plugins>
    </build>

</project>
```

2.3) 配置启动文件

```java
@SpringBootApplication
@MapperScan(basePackages = "com.zelin.mapper")    //此注解用于扫描mapper接口
public class ShiroApplication {
    public static void main(String[] args) {
        SpringApplication.run(ShiroApplication.class);
    }
}
//注意：此启动文件定义在com.zelin这个父包下，目的是可以读取此包及其子包下的所有注解，可以将这些java类的实例放到spring容器中。
```

2.4) 配置application.yml文件或application.properties文件

```yaml
server:
  port: 9000
#配置数据源设置
spring:
  datasource:
    driver-class-name: com.mysql.jdbc.Driver
    url: jdbc:mysql://localhost:3306/shiro
    username: root
    password: 123
#配置mybatis的内容
mybatis:
  mapper-locations: mapper/*.xml
```

2.5) 如何切换数据源

2.5.0) 在pom.xml文件中添加对druid的依赖

```xml
<!--8.添加druid连接池-->
        <dependency>
            <groupId>com.alibaba</groupId>
            <artifactId>druid</artifactId>
            <version>1.0.14</version>
        </dependency>
```

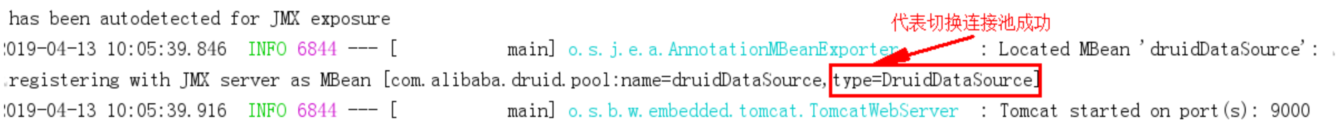
2.5.1) 定义druid.properties文件

```
#配置数据源设置(改变数据源为druid)
spring.datasource.driver-class-name=com.mysql.jdbc.Driver
spring.datasource.url=jdbc:mysql://localhost:3306/shiro
spring.datasource.username=root
spring.datasource.password=123
```

2.5.2) 定义配置类: 在com.zelin.config包下DruidConfiguration.java

```java
/**
 * @Author: Feng.Wang
 * @Company: Zelin.ShenZhen
 * @Description:
 * @Date: Create in 2019/4/13 09:55
 */
@Configuration   //此注解就是相当于原来的applicationContext.xml文件
@PropertySource("classpath:druid.properties")
public class DruidConfiguration {
    @Bean        //此注解：相当于spring配置文件中的<bean>标签
    @ConfigurationProperties(prefix = "spring.datasource")
    //工作原理：就是将druid.properties文件中的前缀后的字符串取出在DruidDataSource类中找以前面
取出的字符串的属性，并为其
    //赋值
    public DruidDataSource druidDataSource(){
        return new DruidDataSource();
    }
}
```

效果如下所示：



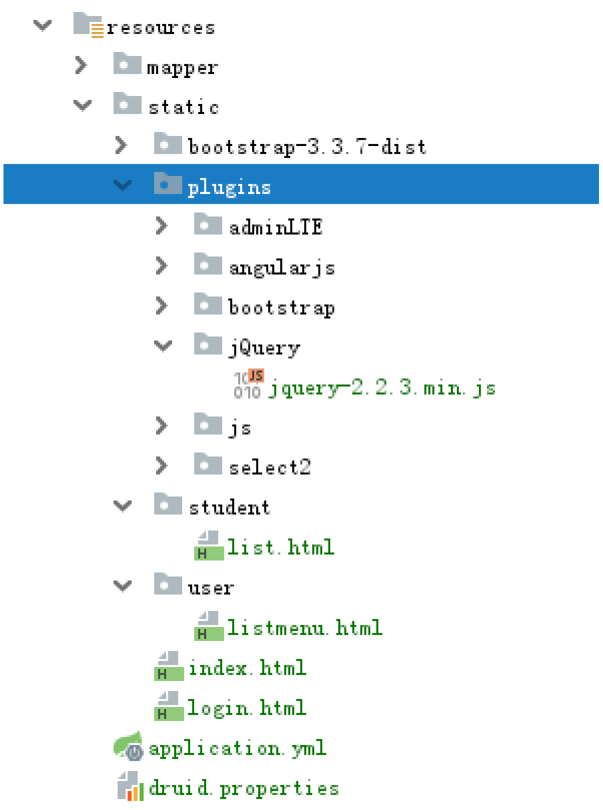
```
 has been autodetected for JMX exposure
019-04-13 10:05:39.846  INFO 6844 --- [           main] o.s.j.e.a.AnnotationMBeanExporter        : Located MBean 'druidDataSource':
registering with JMX server as MBean [com.alibaba.druid.pool:name=druidDataSource,type=DruidDataSource]
019-04-13 10:05:39.916  INFO 6844 --- [           main] o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 9000
```

3、完成上章的用户认证功能

3.1) 使用angular+BootStrap完成登录功能 ：

实现步骤：

1. **添加各种静态资源，如下图所示：**



```
resources
    mapper
    static
        bootstrap-3.3.7-dist
        plugins
            adminLTE
            angularjs
            bootstrap
            jQuery
                jquery-2.2.3.min.js
            js
            select2
        student
            list.html
        user
            listmenu.html
        index.html
        login.html
    application.yml
    druid.properties
```

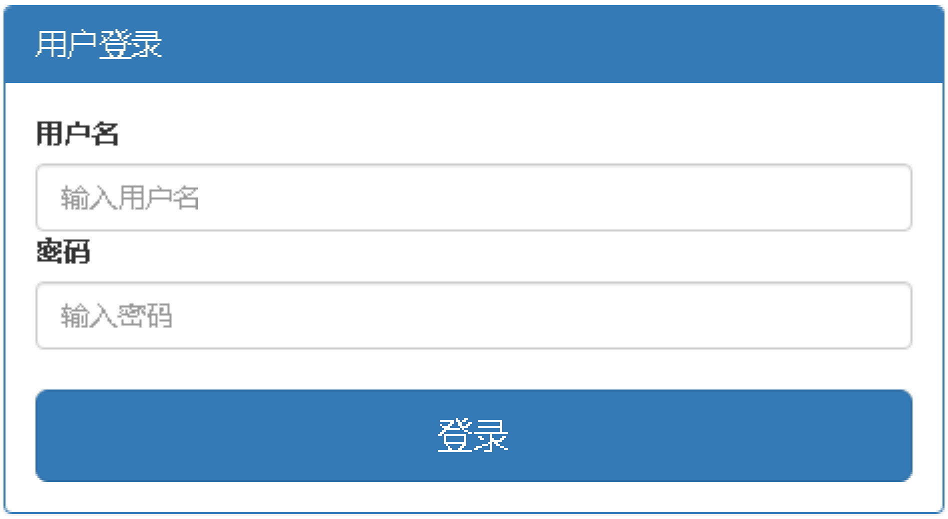
2. **定义login登录页面，内容如下：**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>用户登录</title>
    <!--引入bootstrap样式-->
    <link rel="stylesheet" href="bootstrap-3.3.7-dist/css/bootstrap.min.css">
    <!--引入分页的样式-->
    <link rel="stylesheet" href="plugins/angularjs/pagination.css">
    <script src="plugins/jQuery/jquery-2.2.3.min.js"></script>
    <script src="bootstrap-3.3.7-dist/js/bootstrap.js"></script>
    <!--1.引入angularjs的库-->
    <script src="plugins/angularjs/angular.min.js"></script>
    <!--引入angularjs的分页库-->
    <script src="plugins/angularjs/pagination.js"></script>
    <style>
        .container{
            width: 500px;
            margin-top: 50px;
        }
        .form-signin{
            padding:5px;
        }
        .btn{
            margin-top: 20px;
        }
        .error{
            color:red;
        }
    </style>

    <script>
        //1.定义angularjs的模块：
        var app = angular.module("myApp",[]);
        //2.定义控制器
        app.controller("loginController",function($scope,$http){
            //2.1)查询所有的学生
            $scope.login = ()=>{
                $http.get("login?username="+$scope.username+"&password="+$scope.password).success(response=>{
                    if (response.status){    //代表登录成功
                        location.href = "./user/listmenu.html"
                    } else{
                      $scope.message = response.message;
                    }
                })
            }
        })
    </script>
</head>
<body ng-app="myApp" ng-controller="loginController">
<div class="container">
    <div class="panel panel-primary">
        <div class="panel-heading">
            <h3 class="panel-title">
                用户登录
            </h3>
        </div>
        <div class="panel-body">
            <label>用户名</label>
            <input type="text"  ng-model="username" class="form-control" placeholder="输入用户名" required autofocus>
            <label >密码</label>
            <input type="password" ng-model="password" class="form-control" placeholder="输入密码" required>
            <button class="btn btn-lg btn-primary btn-block" type="button" ng-click="login()">登录</button>
            <span class="error">{{message}}</span>
        </div>
    </div>
</div> <!-- /container -->
</body>
</html>

```

3）定义登录的控制器login

```
@RestController
public class LoginController {
    @Autowired
    private SysUserService userService;
    @RequestMapping("/login")
    public AjaxResult login(String username, String password, HttpServletRequest
request, HttpServletResponse response){
        try {
            //1.根据用户名在数据库中查询是否存在此用户
            SysUser user = userService.findUserByUsercodeAndPassword(username,
password);
            //2.判断用户是否存在
            //2.1)如果此用户存在，就将其放到session中
            if(user != null){
                request.getSession().setAttribute("user",user);
                return new AjaxResult("登录成功",true);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
        return new AjaxResult("登录失败",false);
    }
}
```

4.登录界面如下：



3.2) 完成拦截器定义：

```
@Component
public class AuthticationInterceptor extends HandlerInterceptorAdapter {
    @Override
    public boolean preHandle(HttpServletRequest request, HttpServletResponse response,
Object handler) throws Exception {
        //1.情况一：如果是匿名用户，就放行
        //1.1) 得到当前登录的用户的url地址
        String uri = request.getRequestURI();
        //1.2) 对uri地址作处理
        uri = CommUtils.getPath(uri);
        //1.3) 得到匿名用户的地址列表
        List<String> anonymousURL = ResourcesUtil.gekeyList("anonymousURL");
        //1.4) 判断当前登录的url地址是否在上面的地址集合中,如果在就放行
```

```java
        if (anonymousURL.contains(uri)) return true;

        //情况二：看用户是否登录，如果登录就看其访问地址是否在公用的url地址列表中，
        //2.1) 得到session，并且取得当前登录的用户对象user
        SysUser user = (SysUser) request.getSession().getAttribute("user");
        //2.2) 判断是否存在此用户
        if (null != user) {    //代表用户登录过
            //2.3）读取公用的用户列表(只有登录成功的用户才能访问此列表)
            List<String> commonURL = ResourcesUtil.gekeyList("commonURL");
            //2.4) 判断当前登录用户的访问url地址是否在commonURL中，是就放行
            if (commonURL.contains(uri)) return true;

            //情况三：根据当前用户，得到其访问资源的权限列表
            //3.1) 得到当前登录成功的用户的所有权限列表
            List<SysPermission> permissions = user.getPermissions();
            //3.2) 根据此权限列表遍历出每个权限，再看当前用户的访问url地址是否在其中
            for (SysPermission permission : permissions) {
                if (permission.getUrl().contains(uri)) {
                    return true;
                }
            }
        }
        //其它情况：如果上面的情况都不成立，就跳转到无权访问页面。
      request.getRequestDispatcher("/resure.html").forward(request,response);
        return false;
    }
}

```

3.3) 配置拦截器：

```java
@Configuration
public class MyConfiguration implements WebMvcConfigurer {
    @Autowired
    private AuthticationInterceptor authticationInterceptor;
    //注册拦截器（springboot拦截器与springmvc不一样，还会拦截静态资源 ）
    @Override
    public void addInterceptors(InterceptorRegistry registry) {

 registry.addInterceptor(authticationInterceptor).addPathPatterns("/student/**","/user/**")
                .excludePathPatterns("/bootstrap-3.3.7-dist/**",
                        "/plugins/**","/**/*.html");

    }
}
```

3.4)定义学生列表功能

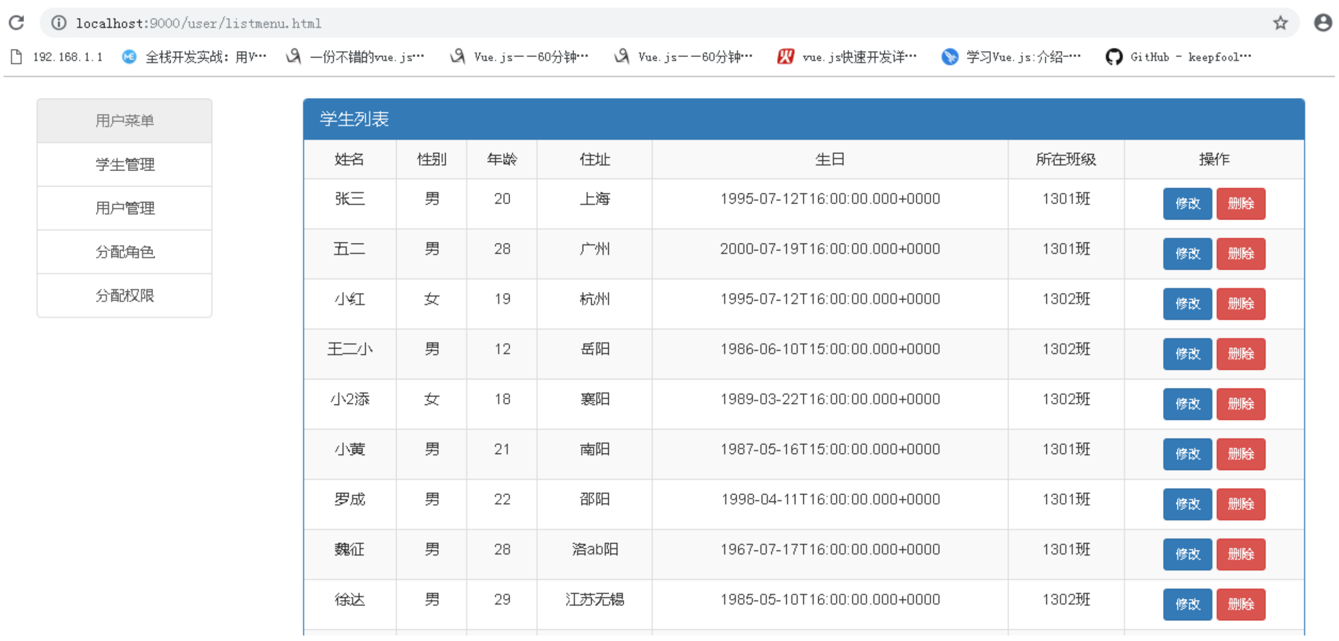
```java
public class StudentController {
    @Autowired
    private StudentService studentService;
    @Autowired
    private ClassesService classesService;
    private int pageSize = 5;
    @RequestMapping("/list")
    public List<Student> findAll(){
        try {
            return studentService.findStudents();
        } catch (Exception e) {
            e.printStackTrace();
            return null;
        }
    }
}
```

```html
/student/list.html---》页面
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>列表学生</title>
    <style>
        .table{
            text-align: center;
        }
        .container{
            margin-top: 20px;
        }
    </style>
    <!--引入bootstrap样式-->
    <link rel="stylesheet" href="../bootstrap-3.3.7-dist/css/bootstrap.min.css">
    <!--引入分页的样式-->
    <link rel="stylesheet" href="../plugins/angularjs/pagination.css">
    <script src="../plugins/jQuery/jquery-2.2.3.min.js"></script>
    <script src="../bootstrap-3.3.7-dist/js/bootstrap.js"></script>
    <!--1.引入angularjs的库-->
    <script src="../plugins/angularjs/angular.min.js"></script>
    <!--引入angularjs的分页库-->
    <script src="../plugins/angularjs/pagination.js"></script>
    <script>
        //1.定义angularjs的模块：
        var app = angular.module("myApp",[]);
        //2.定义控制器
        app.controller("studentController",function($scope,$http){
           //2.1)查询所有的学生
            $scope.findAll=()=>{
                $http.get("../student/list").success(response=>{
                    //1.为list变量分配值
                    $scope.list = response;
                } )
            }
        })
    </script>
</head>
```

```
<body ng-app="myApp" ng-controller="studentController" ng-init="findAll()">
<div class="container">
    <div class="panel panel-primary">
        <div class="panel-heading">
            <h3 class="panel-title">学生列表</h3>
        </div>
        <table class="table table-bordered table-striped">
            <tr>
                <td>姓名</td>
                <td>性别</td>
                <td>年龄</td>
                <td>住址</td>
                <td>生日</td>
                <td>所在班级</td>
                <td>操作</td>
            </tr>
                <tr ng-repeat="stud in list">
                    <td>{{stud.sname}}</td>
                    <td>{{stud.sex}}</td>
                    <td>{{stud.age}}</td>
                    <td>{{stud.addr}}</td>
                    <td>
                        {{stud.birth}}
                    </td>
                    <td>
                        {{stud.classes.cname}}
                    </td>
                    <td>
                        <a class="btn btn-primary btn-sm"
                           href="../student/toupdate?sid=${stud.sid}">修改</a>
                        <a class="btn btn-danger btn-sm"
                           href="../student/deleteBySid?sid=${stud.sid}"
                           onclick="return confirm('你真的要删除吗?')">删除</a>
                    </td>
                </tr>
        </table>
    </div>
</div>
<script>
    //执行提交 表单
    function skip(i) {
        //1.对表单中的隐藏域赋值
        $("#page").val(i);
        //2.提交表单
        $("#form1").submit();
    }

    //添加学生
    function addStudent() {
        location.href = "${pageContext.request.contextPath}/student/toadd.do";
    }
</script>
</body>
```

```
</html>
```

3.5）效果图如下：



学生列表

姓名	性别	年龄	住址	生日	所在班级	操作
张三	男	20	上海	1995-07-12T16:00:00.000+0000	1301班	修改 删除
五二	男	28	广州	2000-07-19T16:00:00.000+0000	1301班	修改 删除
小红	女	19	杭州	1995-07-12T16:00:00.000+0000	1302班	修改 删除
王二小	男	12	岳阳	1986-06-10T15:00:00.000+0000	1302班	修改 删除
小2添	女	18	襄阳	1989-03-22T16:00:00.000+0000	1302班	修改 删除
小黄	男	21	南阳	1987-05-16T15:00:00.000+0000	1301班	修改 删除
罗成	男	22	邵阳	1998-04-11T16:00:00.000+0000	1301班	修改 删除
魏征	男	28	洛ab阳	1967-07-17T16:00:00.000+0000	1301班	修改 删除
徐达	男	29	江苏无锡	1985-05-10T16:00:00.000+0000	1302班	修改 删除

4、完成为当前是用户修改角色的功能：

4.1）在RoleController控制器得到所有的角色列表：

```java
@RestController
@RequestMapping("/role")
public class RoleController {
    @Autowired
    private RoleService roleService;
    @Autowired
    private UserRoleService userRoleService;
    /**
     * 查询所有的角色
     * @return
     */
    @RequestMapping("/list")
    public List<SysRole> findAll(){
        try {
            return roleService.findRoles();
        } catch (Exception e) {
            e.printStackTrace();
        }
        return  null;
    }
}
```

4.2）根据当前登录用户查询出其关联的角色

```java
 /**
     * 根据当前登录用户查询出其关联的角色
     * @param session
     * @return
     */
    @RequestMapping("/findRolesByUser")
    public List<SysUserRole> findRolesByUser(HttpSession session){
        Object object = session.getAttribute("user");
        if(object != null){
            SysUser user = (SysUser) object;
            List<SysUserRole> userRoles =
userRoleService.findUserRole(user.getUsercode());
            return userRoles;
        }
        return null;

    }
```

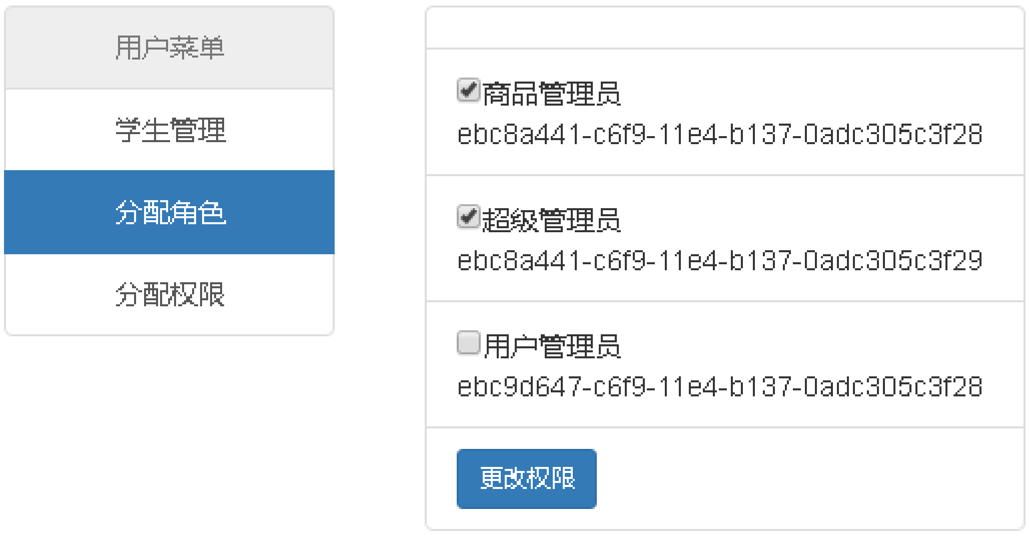
4.3) 修改当前用户的角色

```
 /**
     * 修改权限
     * @param sysUserRoleVo
     * @param session
     * @return
     */
    @RequestMapping("/changeRole")
    public AjaxResult changeRole(SysUserRoleVo sysUserRoleVo,HttpSession session){
        try {
            SysUser user = (SysUser) session.getAttribute("user");
            if(user != null){
                sysUserRoleVo.setSysUserId(user.getId());
            }
            userRoleService.update(sysUserRoleVo);
            return new AjaxResult("修改权限成功",true);
        } catch (Exception e) {
            e.printStackTrace();
            return new AjaxResult("修改权限失败",false);
        }
    }
```

4.4) user/changeRole.html页面的内容

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>修改用户权限</title>
    <!--引入bootstrap样式-->
    <link rel="stylesheet" href="../bootstrap-3.3.7-dist/css/bootstrap.min.css">
    <!--引入分页的样式-->
    <link rel="stylesheet" href="../plugins/angularjs/pagination.css">
    <script src="../plugins/jQuery/jquery-2.2.3.min.js"></script>
    <script src="../bootstrap-3.3.7-dist/js/bootstrap.js"></script>
    <!--1.引入angularjs的库-->
    <script src="../plugins/angularjs/angular.min.js"></script>
    <!--引入angularjs的分页库-->
    <script src="../plugins/angularjs/pagination.js"></script>
    <style>
        .list-group{
            width: 300px;
            margin-top: 20px;
            margin-left: 40px;
        }
    </style>
    <script>
        //1.定义angularjs的模块：
        var app = angular.module("myApp",[]);
        //2.定义控制器
        app.controller("roleController",function($scope,$http){
            //2.1)查询所有的权限
            $scope.findAll=()=>{
                $http.get("../role/list").success(response=>{
                    $scope.list = response;

                    findRolesByUser();
                })
            }
            //2.2)根据当前用户取出其角色列表
            findRolesByUser = ()=>{
                $http.get("../role/findRolesByUser").success(response =>{
                    $scope.roleList = response;
                    //遍历$scope.roleList与$scope.list两个集合，选中复选框
                    if ($scope.list.length > 0 && $scope.roleList.length > 0)
                    for (var i = 0;i  < $scope.list.length;i++){
                        $scope.list[i]["result"] = false;
                        var role = $scope.list[i];
                        for (var j = 0; j < response.length;j++){
                            var roleUser = response[j];
                            if (role.id ==roleUser.sysRoleId){
                                $scope.list[i]["result"] = true;
                                //如果两个id相等先放到角色id数组中
                                $scope.sysRoleIds.push(role.id)
                            }
                        }
                    }
                } )
            }
            //定义所选择的角色id数组
            $scope.sysRoleIds = [];
            $scope.selectRole=(roleId,event)=>{
                if(event.target.checked){   //如果选择了某个角色就将其放到角色id数组中
                    $scope.sysRoleIds.push(roleId);
                }else{                      //否则，就从角色id数组中删除它
                    $scope.sysRoleIds.splice($scope.sysRoleIds.indexOf(roleId),1)
                }
            }
            //3.修改角色
            $scope.changeRole = ()=>{

                //将角色id数组提交到后台
               $http.get("../role/changeRole?sysRoleIds="+$scope.sysRoleIds).success(response=>{
                    if (!response.status){   //修改失败
                        alert(response.message);
                    }
                })
            }
        })
     </script>

</head>
<body ng-app="myApp" ng-controller="roleController" ng-init="findAll()">
{{sysRoleIds}}---

        <ul class="list-group">
            <li class="list-group-item" >{{name}}
            <li class="list-group-item" ng-repeat="role in list ">
            {{role.id}}
               <!--<input type="checkbox"   ng-checked="role.result" ng-model="sysRoleIds[$index]"-->
                      <!--ng-true-value="{{role.id}}" ng-false-value="">{{role.name}}-->
                    <input type="checkbox"   ng-checked="role.result" ng-click="selectRole(role.id,$event)"
                        ng-true-value="{{role.id}}" ng-false-value="">{{role.name}}
            </li>
            <li class="list-group-item" >
                <input type="button" value="更改权限" class="btn btn-primary btn-sm" ng-click="changeRole()">
            </li>
        </ul>
</body>
</html>
```

4.5) 运行结果如下：



☑商品管理员
ebc8a441-c6f9-11e4-b137-0adc305c3f28

☑超级管理员
ebc8a441-c6f9-11e4-b137-0adc305c3f29

☐用户管理员
ebc9d647-c6f9-11e4-b137-0adc305c3f28