Apache Shiro权限框架实战+项目案例

注意：代码和资料在视频课程最后几个章节。
请在电脑浏览器上进行下载！

主讲：安燚

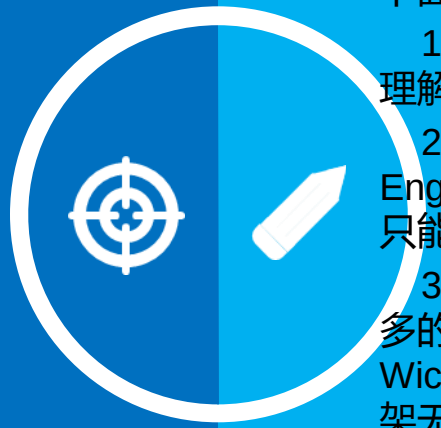
1 shiro简介

Apache Shiro是什么

Apache Shiro 是功能强大并且容易集成的开源权限框架，它能够完成认证、授权、加密、会话管理、与Web集成、缓存等。认证和授权为权限控制的核心，简单来说，

“认证”就是证明你是谁？ Web 应用程序一般做法通过表单提交用户名及密码达到认证目的。

“授权”即是否允许已认证用户访问受保护资源。



为何对 Shiro 情有独钟

Spring Security和Shiro ？

下面对两者略微比较：

1、简单性，Shiro 在使用上较 Spring Security 更简单，更容易理解。适合于入门。

2、灵活性，Shiro 可运行在 Web、EJB、IoC、Google App Engine 等任何应用环境，却不依赖这些环境。而 Spring Security 只能与 Spring 一起集成使用。

3、可插拔，Shiro 干净的 API 和设计模式使它可以方便地与许多的其它框架和应用进行集成。Shiro 可以与诸如 Spring、Grails、Wicket、Tapestry、Mule、Apache Camel、Vaadin 这类第三方框架无缝集成。

Spring Security 在这方面就显得有些捉衿见肘。

1 介绍-系统架构

系统架构

核心框架：Spring Framework 4.2

安全框架：Apache Shiro 1.3

视图框架：Spring MVC 4.2

持久层框架：MyBatis 3.3

定时器：Quartz 2.2

数据库连接池：Druid 1.0

日志管理：SLF4J 1.7、Log4j

页面交互：Vue2.x

注：只针对shiro部分进行了详解，其他技术不会讲解。

1 介绍-系统特点

该项目案例是一个轻量级权限管理系统，其核心设计目标是开发迅速、学习简单、轻量级、易扩展等。

特点如下：

1、轻量级的权限系统，只涉及Spring、Shiro、Mybatis后端框架，降低学习使用成本

2、友好的代码结构及注释，便于阅读及二次开发

3、灵活的权限控制，可控制到页面或按钮

4、页面交互使用Vue2.x，极大的提高了开发效率（支持HTML、JSP、Velocity、Freemarker等视图）

5、完善的代码生成机制，可在线生成entity、xml、dao、service、page、js代码，3分钟可以完成一个简单的增删改查页面。

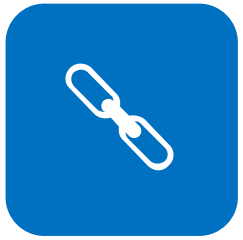
6、引入quartz定时任务，可动态完成任务的添加、修改、删除、暂停、恢复及日志查看等功能

第二部分
Shiro项目案例分解

1. 环境搭建
2. 框架集成
3. 权限设计及实现
4. 菜单管理
5. 角色管理
6. 用户管理
7. 定时任务设计及实现
8. 快速生成代码设计及实现
9. 其他

主讲：安燚

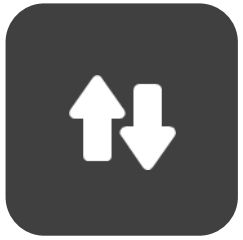
1 环境准备



JDK1.7+

Maven3.0+

MySQL5.5+



Tomcat8.0+

STS (eclipse)

其他

1 项目部署

01 建库

创建数据库shiro，数据库编码为UTF-8
执行src/main/resources/db.sql文件，初始化数据
修改db.properties文件，更新MySQL账号和密码

02 编译更新

Eclipse、IDEA执行【clean package tomcat7:run】命令，即可运行项目

03 访问

项目访问路径：http://localhost

非Maven方式启动，则默认访问路径为：
http://localhost:8080/shiro
默认管理员：admin/admin

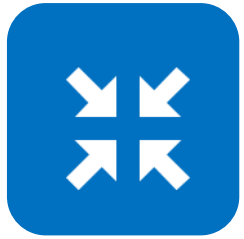
建议使用阿里云Maven仓库

```
<mirror>
<id>alimaven</id>
<name>aliyun maven</name>
<url>http://maven.aliyun.com/nexus/content/groups/public/</url>
<mirrorOf>central</mirrorOf>
</mirror>
```

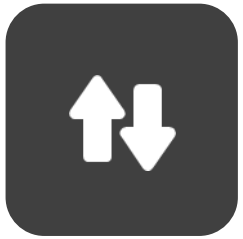
2 框架集成

spring.xml

spring-shiro.xml



spring-scheduler.xml



spring-mvc.xml

spring-jdbc.xml

mybatis.xml

log4j.properties

3 权限设计及实现

1、权限？

权限管理往往是一个极其复杂的问题，但也可简单表述为这样的逻辑表达式：判断“Who对What(Which)进行How的操作”的逻辑表达式是否为真。

针对不同的应用，需要根据项目的实际情况和具体架构，在维护性、灵活性、完整性等N多个方案之间比较权衡，选择符合的方案。

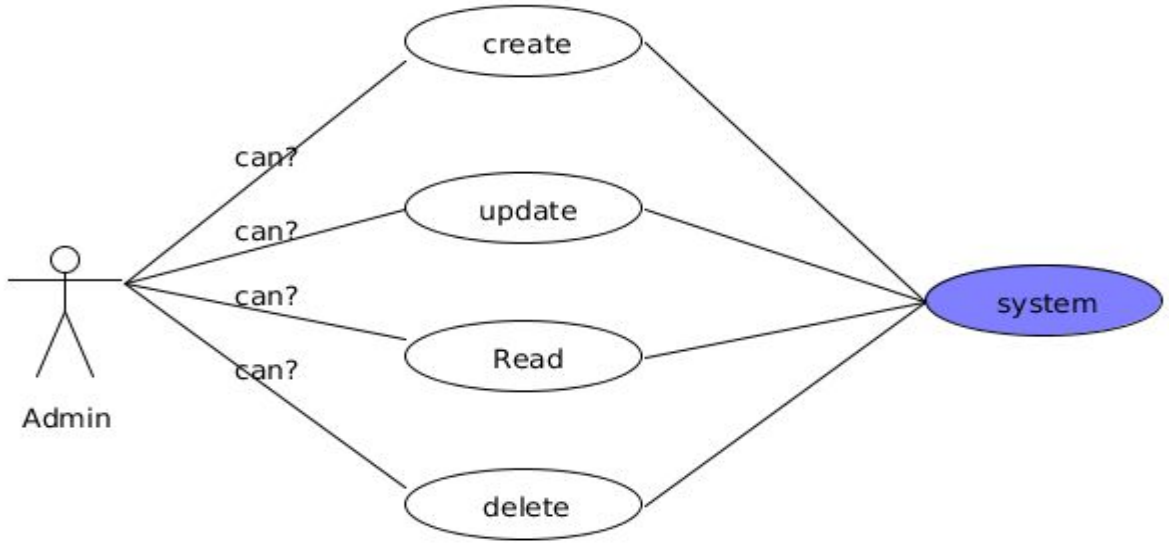
2、目标：

直观，因为系统最终会由最终用户来维护，权限分配的直观和容易理解，显得比较重要简单，

包括概念数量上的简单和意义上的简单还有功能上的简单。

想用一个权限系统解决所有的权限问题是不现实的。

设计中将常常变化的“定制”特点比较强的部分判断为业务逻辑，而将常常相同的“通用”特点比较强的部分判断为权限逻辑就是基于这样的思路。



3 权限设计及实现

3、思想：

权限系统的核心由以下三部分构成：1.创造权限，2.分配权限，3.使用权限，

然后，系统各部分的主要参与者对照如下：

1.创造权限 - Creator创造，一个子系统或称为模块，应该有哪些权限。

2.分配权限 - Administrator 分配，如，创建角色，创建用户组，给用户组分配用户，将用户组与角色关联等等...这些操作都是由 Administrator 来完成的。

3.使用权限 - User：就是什么权限可以访问什么资源

3 权限设计及实现

4、常见的角色模型：

A一般简单的角色模型为：用户 -> 权限

B最常用的角色模型为：用户-〉角色-〉权限

C较复制的关系模型包括了用户、部门、角色、权限、模块

3 权限设计及实现

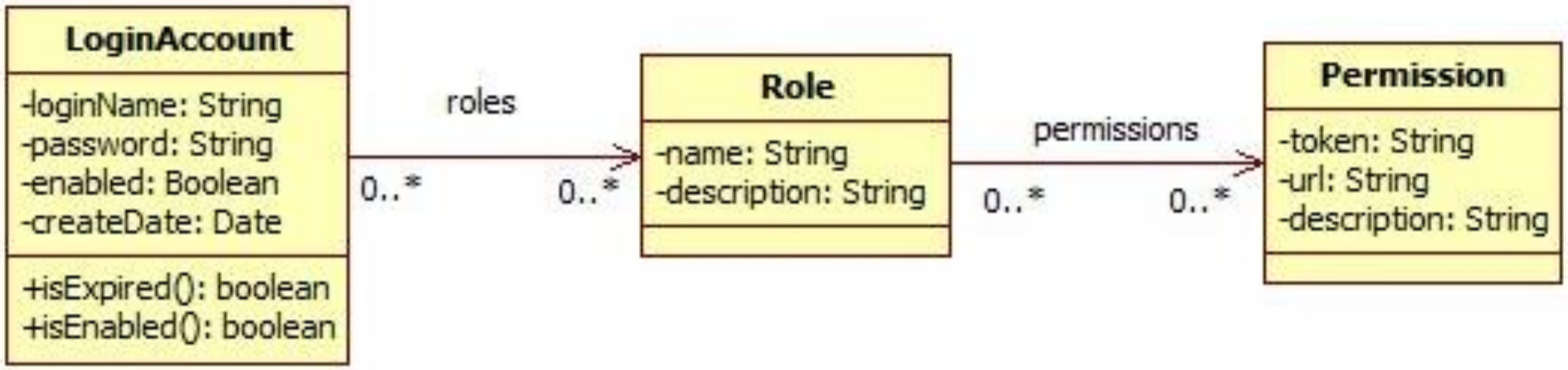
认识用户权限模型：

这里所提到用户权限模型，指的是用来表达用户信息及用户权限信息的数据模型。即能证明“你是谁？”、“你能访问多少受保护资源？”。为实现一个较为灵活的用户权限数据模型，通常把用户信息单独用一个实体表示，用户权限信息用两个实体表示。

1、用户信息用 LoginAccount 表示，最简单的用户信息可能只包含用户名 loginName 及密码 password 两个属性。实际应用中可能会包含用户是否被禁用，用户信息是否过期等信息。

2、用户权限信息用 Role 与 Permission 表示，Role 与 Permission 之间构成多对多关系。Permission 可以理解为对一个资源的操作，Role 可以简单理解为 Permission 的集合。

3、用户信息与 Role 之间构成多对多关系。表示同一个用户可以拥有多个 Role，一个 Role 可以被多个用户所拥有。



3 权限设计及实现

数据库设计：

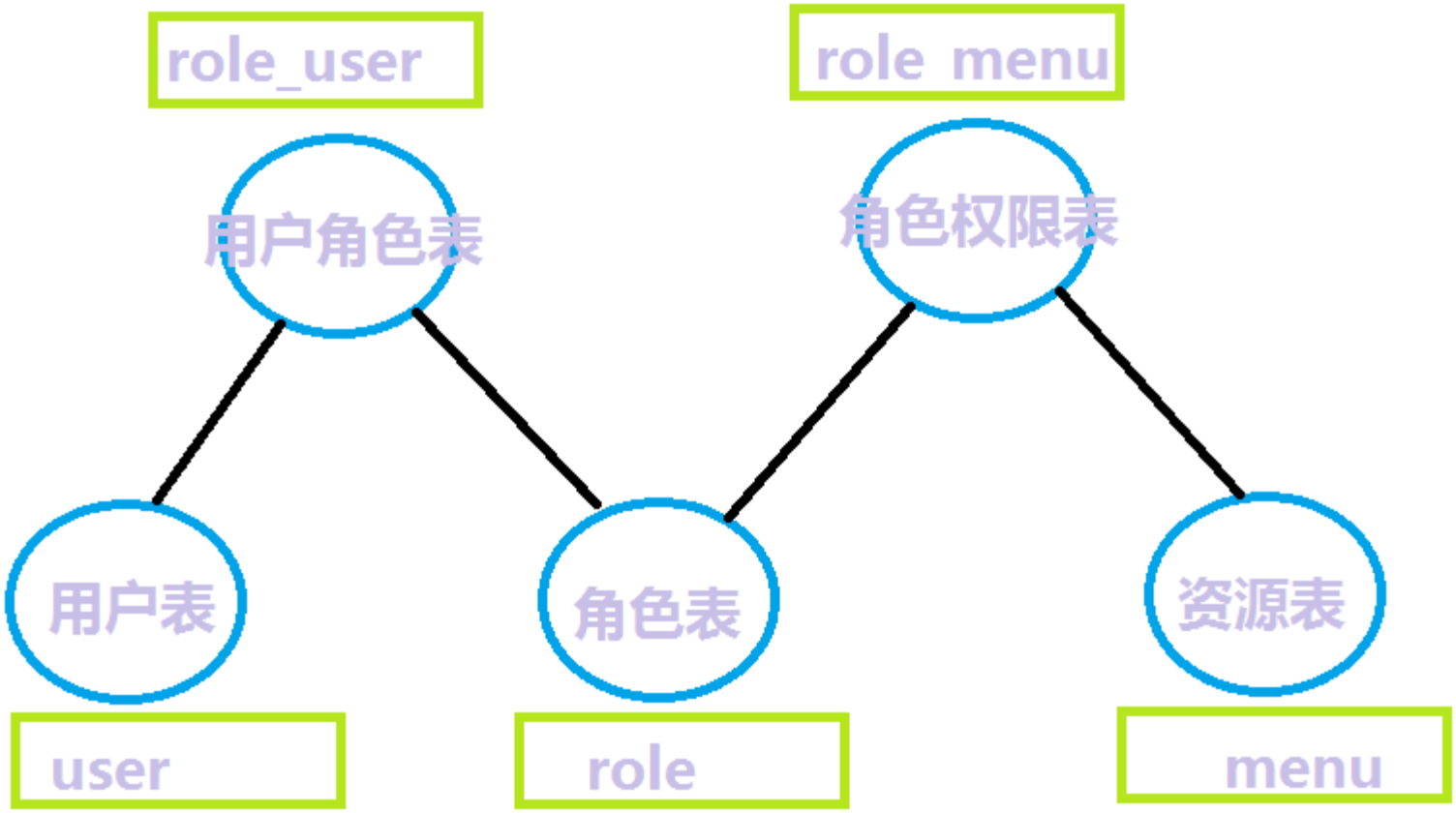
sys_menu定义目录、菜单、按钮（权限许可范围）

sys_role定义角色信息

sys_role_menu存放角色与菜单对应关系

sys_user定义系统用户

sys_user_role用户与角色对应关系



4 菜单管理

管理系统的目录、菜单、按钮资源，提供新增、修改、删除功能。

目录：指的是一个抽拉项目，即【系统管理】
菜单：目录下的菜单项，即导航栏【角色列表】
按钮：页面的具体功能，新增、修改、删除、暂停、恢复等

添加顺序：
目录→菜单→按钮

相关表：
sys_menu

4 菜单管理

注意：
【权限标识一定要在这里添加，不然没权限访问接口；多个权限的情况，用逗号,分隔】
如：【管理员列表】：
查看-->sys:schedule:list,sys:schedule:info
新增-->sys:schedule:save
修改-->sys:schedule:update
删除-->sys:schedule:delete
暂停-->sys:schedule:pause
恢复-->sys:schedule:resume
立即执行-->sys:schedule:run
日志列表-->sys:schedule:log

5 角色管理

定义系统角色，并且对访问资源进行授权。提供新增、修改、删除功能。

相关表：
sys_role
sys_role_menu

6 用户管理

管理系统人员帐号信息，并提供新增、修改、删除功能。帐号启用、禁用。
可以绑定系统角色。

关联表：
sys_user
sys_user_role

7 定时任务-设计原理

定时任务使用场景还是比较多的，一般项目都会使用到定时任务，

如：定时发送短信、邮件、商品定时上架、下架、优惠券过期、未支付订单取消、快递信息更新、数据报表、数据统计、结算等等；

常用定时任务设计：

1、使用timer

2、使用spring schedule

3、使用quartz

4、使用其他基于quartz的插件

相关的建表语句：

quartz-2.2.3\docs\dbTables

本项目使用Quartz2.2.x实现定时任务功能，支持添加、修改、删除、暂停、恢复、集群及日志查看等功能。

默认支持集群。

7 定时任务-实战演练

1、新增定时任务，只需创建spring bean即可，如下所示：

```
/**
* 新增定时任务
*
* bean的名称为【newTask】
*/
@Component("newTask")
public class NewTask {
        private Logger logger = LoggerFactory.getLogger(getClass());
        public void test1(String params){
                logger.info("我是带参数的test1方法，正在被执行，参数为：" + params);
        }
        public void test2(){
                logger.info("我是不带参数的test2方法，正在被执行");
        }
}
```

2、在管理后台，添加定时任务，如下图所示：

bean名称：newTask
方法名称：test1
参数：shiro
cron表达式：0 0 12 * * ?
备注：测试

完成上面2步，新的定时任务就添加完成了，每天12点都会执行一次

7 定时任务-实战演练

表 5.1. Quartz Cron 表达式支持到七个域

名称	是否必须	允许值	特殊字符
秒	是	0-59	, - * /
分	是	0-59	, - * /
时	是	0-23	, - * /
日	是	1-31	, - * ? / L W C
月	是	1-12 或 JAN-DEC	, - * /
周	是	1-7 或 SUN-SAT	, - * ? / L C #
年	否	空 或 1970-2099	, - * /

Eg:
分钟频度的任务计划 Cron 表达式:
每天的从 5:00 PM 至 5:59 PM 中的每分钟触发 [0 * 17 * * ?]

日的频度的任务计划 Cron 表达式:
每天的 3:00 AM [0 0 3 * * ?]

周和/或月的频度上任务计划的 Cron 表达式:
在每个周一,二, 三和周四的 10:15 AM [0 15 10 ? * MON-FRI]
每月15号的 10:15 AM [0 15 10 15 * ?]

8 快速生成代码-设计原理

代码生成器使用

代码生成器是根据表结构，自动生成相应的代码，需先在MySQL中建好表结构，再使用代码生成器

代码生成器是通过velocity模板实现的，依赖velocity所需jar包，可在线生成entity、xml、dao、service、page、js代码的zip压缩文件

修改包名、作者、作者邮箱，需在generator.properties 中配置

如需去掉表tb_user 前缀tb_ ，可配置tablePrefix 项，如： tablePrefix=tb_ ，则生成的实体类为UserEntity ，否则生成TbUserEntity

可根据自己的需求，自行修改模板，模板代码位置：【resources\template】

模板数据封装在如下所示map里，其中tableEntity 对象为TableEntity.java 实例， config 为generator.properties 配置文件数据

8 快速生成代码-设计原理

```
//模板数据
Map<String, Object> map = new HashMap<>();
map.put("tableName", tableEntity.getTableName());
map.put("comments", tableEntity.getComments());
map.put("pk", tableEntity.getPk());//数据库主键，没有则为第一个字段
map.put("className", tableEntity.getClassName());
map.put("classname", tableEntity.getClassname());
map.put("pathName", tableEntity.getClassname().toLowerCase());
map.put("columns", tableEntity.getColumns());
map.put("package", config.getString("package"));
map.put("author", config.getString("author"));
map.put("email", config.getString("email"));
map.put("datetime", DateUtils.format(new Date(), DateUtils.DATE_TIME_PATTERN));
```

MySQL数据类型与Java数据类型转换，在generator.properties 中配置，如有些类型的转换关系不存在，则需在generator.properties 中添加，如： bigint=Long

生成的html、js代码，需要修改html代码里的js路径，避免404错误

添加相应的菜单即可【权限标识一定要添加，不然没权限访问接口】

8 快速生成代码-实战演练

演示快速开发一个模块步骤：

1、手动创建一张表（需要添加comments）

2、登录系统并且使用自动生成代码模块

3、将生成的java代码放入对应的工程目录中

4、将生成的js文件放入对应工程目录中

5、将生成的html文件放入对应工程目录中，并修改加载js文件目录路径

6、使用系统中的[菜单管理]-->新增菜单-->sys/config.html

7、使用系统中的[菜单管理]-->新增按钮-->依次添加按钮【新增】、【修改】、【删除】、【查看】并且进行授权，多个权限用逗号,隔开

9 其他-视图支持

多视图使用

默认是支持JSP、Velocity、Freemarker视图

文件后缀以jsp结尾【如：test.jsp】，则会使用JSP视图

文件后缀以html结尾【如：test.html】，则会使用Velocity视图

文件后缀以f tl结尾【如：test.f tl】，则会使用Freemarker视图

本系统是通过if rame方式嵌入页面的，可以使用任何前端框架

9 其他-异常处理

异常处理

系统做了异常统一处理，无需try catch，只需往外抛就行，异常处理如下：

```
@Component
public class RRExceptionHandler implements HandlerExceptionResolver {
                        private Logger logger = LoggerFactory.getLogger(getClass());
                        @Override
                        public ModelAndView resolveException(HttpServletRequest request,
                        HttpServletResponse response, Object handler, Exception ex) {
                        R r = new R();
                        try {
                        response.setContentType("application/json;charset=utf-8");
                        response.setCharacterEncoding("utf-8");
                        if (ex instanceof RRException) {
                        r.put("code", ((RRException) ex).getCode());
                        r.put("msg", ((RRException) ex).getMessage());
                        }else if(ex instanceof DuplicateKeyException){
                        r = R.error("数据库中已存在该记录");
                        }else if(ex instanceof AuthorizationException){
                        r = R.error("没有权限，请联系管理员授权");
                        }else{
                        r = R.error();
                        }
                        //记录异常日志
                        logger.error(ex.getMessage(), ex);
                        String json = JSON.toJSONString(r);
                        response.getWriter().print(json);
                        } catch (Exception e) {
                        logger.error("RRExceptionHandler 异常处理失败", e);
                        }
                        return new ModelAndView();
                        }
}
```

9 其他-参数配置

定义数据字典的维护。并提供新增、删除、修改功能。

关联表：
sys_config

感谢您的参与!

主讲：安燚