

细说 Spring Security 安全框架（三）

3.4 定义用户，角色，角色关系表

用户信息表 sys_user：

对象 sys_user @ssm (127.0.0.1_m...	
新建 保存 另存为 添加字段 插	
字段	索引 外键 触发器 选项 注释 SQL 预览
名	类型
id	int
username	varchar
password	varchar
realname	varchar
isenable	int
islock	int
iscredentials	int
createtime	date
logintime	date

sys_role：角色表

字段	索引 外键 触发器 选项 注释 SQL 预览
名	类型
id	int
rolename	varchar
rolememo	varchar

sys_user_role: 用户-角色关系表

名	类型
userid	int
roleid	int

3.5 pom 依赖

```
<parent>

  <groupId>org.springframework.boot</groupId>

  <artifactId>spring-boot-starter</artifactId>

  <version>2.0.6.RELEASE</version>

</parent>

<dependencies>

  <dependency>

    <groupId>junit</groupId>

    <artifactId>junit</artifactId>

    <version>4.11</version>

    <scope>test</scope>

  </dependency>

  <!--security-->

  <dependency>

    <groupId>org.springframework.boot</groupId>

    <artifactId>spring-boot-starter-security</artifactId>

  </dependency>

  <!--web-->

  <dependency>
```

```
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-web</artifactId>
</dependency>

<!--mysql 驱动-->
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>5.1.9</version>
</dependency>

<!--mybatis-->
<dependency>
    <groupId>org.mybatis.spring.boot</groupId>
    <artifactId>mybatis-spring-boot-starter</artifactId>
    <version>2.0.1</version>
</dependency>
</dependencies>
```

3.6 创建实体类

定义用户信息类，实现 spring security 框架中的 UserDetails.

```
public class SysUser implements UserDetails {
```

```
private Integer id;

private String username;

private String password;

private String realname;


private boolean isExpired;

private boolean isLocked;

private boolean isCredentials;

private boolean isEnabled;


private Date createTime;

private Date loginTime;


private List<GrantedAuthority> authorities;


public SysUser() {

}


public SysUser(String username, String password, String realname,
               boolean isExpired, boolean isLocked,
```

```
        boolean isCredentials, boolean isEnabled,  
        Date createTime, Date  
loginTime, List<GrantedAuthority> authorities) {  
    this.username = username;  
    this.password = password;  
    this.realname = realname;  
    this.isExpired = isExpired;  
    this.isLocked = isLocked;  
    this.isCredentials = isCredentials;  
    this.isEnabled = isEnabled;  
    this.createTime = createTime;  
    this.loginTime = loginTime;  
    this.authorities = authorities;  
}  
  
@Override  
public Collection<? extends GrantedAuthority> getAuthorities() {  
    return authorities;  
}  
  
@Override
```

```
public String getPassword() {  
    return password;  
}
```

@Override

```
public String getUsername() {  
    return username;  
}
```

@Override

```
public boolean isAccountNonExpired() {  
    return isExpired;  
}
```

@Override

```
public boolean isAccountNonLocked() {  
    return isLocked;  
}
```

@Override

```
public boolean isCredentialsNonExpired() {
```

```
        return isCredentials;

    }

    @Override

    public boolean isEnabled() {

        return isEnabled;

    }


    public Integer getId() {

        return id;

    }


    public Date getCreateTime() {

        return createTime;

    }


    public Date getLoginTime() {

        return loginTime;

    }


    public String getRealname() {
```

```
        return realname;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    public void setRealname(String realname) {
        this.realname = realname;
    }

    public void setExpired(boolean expired) {
```



```
        isExpired = expired;
    }

    public void setLocked(boolean locked) {

        isLocked = locked;
    }

    public void setCredentials(boolean credentials) {

        isCredentials = credentials;
    }

    public void setEnabled(boolean enabled) {

        isEnabled = enabled;
    }

    public void setCreateTime(Date createTime) {

        this.createTime = createTime;
    }

    public void setLoginTime(Date loginTime) {

        this.loginTime = loginTime;
    }
}
```

```
}
```

```
public void setAuthorities(List<GrantedAuthority> authorities) {
```

```
    this.authorities = authorities;
```

```
}
```

```
@Override
```

```
public String toString() {
```

```
    return "SysUser{" +
```

```
        "id=" + id +
```

```
        ", username=" + username + '\n' +
```

```
        ", password=" + password + '\n' +
```

```
        ", realname=" + realname + '\n' +
```

```
        ", isExpired=" + isExpired +
```

```
        ", isLocked=" + isLocked +
```

```
        ", isCredentials=" + isCredentials +
```

```
        ", isEnabled=" + isEnabled +
```

```
        ", createTime=" + createTime +
```

```
        ", loginTime=" + loginTime +
```

```
        ", authorities=" + authorities +
```

```
    };
```

```
}  
  
}
```

SysRole 类：

```
public class SysRole {  
  
    private Integer id;  
  
    private String name;  
  
    private String memo;  
  
    public Integer getId() {  
  
        return id;  
  
    }  
  
    public void setId(Integer id) {  
  
        this.id = id;  
  
    }  
  
    public String getName() {  
  
        return name;  
  
    }  
  
}
```

```
public void setName(String name) {  
    this.name = name;  
}  
  
public String getMemo() {  
    return memo;  
}  
  
public void setMemo(String memo) {  
    this.memo = memo;  
}  
  
@Override  
public String toString() {  
    return "SysRole{" +  
        "id=" + id +  
        ", name=" + name + "\" +  
        ", memo=" + memo + "\" +  
        }";  
}  
}
```

3.7 Mapper 类

@Repository //创建 dao 对象

```
public interface SysUserMapper {
```

```
    int insertSysUser(SysUser user);
```

```
    //根据账号名称，获取用户信息
```

```
    SysUser selectSysUser(String username);
```

```
}
```

```
public interface SysRoleMapper {
```

```
    List<SysRole> selectRoleByUser(Integer userId);
```

```
}
```

3.8 创建 UserDetatilsService 接口的实现类

根据 username 从数据查询账号信息 SysUser 对象。

在根据 user 的 id，去查 Sys_Role 表获取 Role 信息

@Service

```
public class JdbcUserDetatilsService implements UserDetailsService {
```

@Autowired

```
private SysUserMapper userMapper;
```

@Autowired

```
private SysRoleMapper roleMapper;
```

@Override

```
public UserDetails loadUserByUsername(String username) throws  
UsernameNotFoundException {
```

```
    //1. 根据 username 获取 SysUser
```

```
    SysUser user = userMapper.selectSysUser(username);
```

```
    System.out.println("loadUserByUsername user:"+user);
```

```
    if( user != null){
```

```
        //2. 根据 userid 的, 获取 role
```

```
        List<SysRole> roleList =
```

```
        roleMapper.selectRoleByUser(user.getId());
```

```
        System.out.println("roleList:"+ roleList);
```

```
List<GrantedAuthority> authorities = new ArrayList<>();

String roleName = "";

for (SysRole role : roleList) {

    roleName = role.getName();

    GrantedAuthority authority = new
SimpleGrantedAuthority("ROLE_"+roleName);

    authorities.add(authority);

}

user.setAuthorities(authorities);

return user;

}

return user;

}

}
```

3.9 创建 Controller

1.IndexController : 转发到首页 index.html

```
@Controller

public class IndexController {
```

```
@GetMapping("/index")

public String toIndexHtml(){

    return "forward:/index.html";

}

}
```

2.MyController

```
@RestController

public class MyController {

    @GetMapping(value = "/access/user",produces =
"text/html;charset=utf-8")

    public String sayUser(){

        return "zs 是 user 角色";

    }

    @GetMapping(value = "/access/read",produces =
"text/html;charset=utf-8")

    public String sayRead(){

        return "lisi 是 read 角色";

    }

}
```



```
@GetMapping(value = "/access/admin", produces =  
"text/html;charset=utf-8")  
  
public String sayAdmin(){  
  
    return "admin 是 user , admin 角色";  
  
}  
  
}
```

3.10 继承 WebSecurityConfigurerAdapter

注入自定义的 UserDetailsService

```
@Configuration  
  
@EnableWebSecurity  
  
public class CustomSecurityConfig extends WebSecurityConfigurerAdapter {  
  
    @Autowired  
  
    private UserDetailsService userDetailsService;  
  
    @Override  
  
    protected void configure(AuthenticationManagerBuilder auth) throws  
Exception {
```

```
//super.configure(auth);

auth.userService(userDetailsService).passwordEncoder(new
BCryptPasswordEncoder());

}

@Override

protected void configure(HttpSecurity http) throws Exception {

    System.out.println("=====configure HttpSecurity===== ");

    http.authorizeRequests()

        .antMatchers("/index").permitAll()

        .antMatchers("/access/user/**").hasRole("USER")

        .antMatchers("/access/read/**").hasRole("READ")

        .antMatchers("/access/admin/**").hasRole("ADMIN")

        .anyRequest().authenticated()

        .and()

        .formLogin();

}

}
```

3.11 主类

```
@MapperScan(value = "com.wkcto.mapper")

@SpringBootApplication

public class UserRoleApplication {

    @Autowired

    SysUserMapper userMapper;

    public static void main(String[] args) {

        SpringApplication.run(UserRoleApplication.class,args);

    }

    // @PostConstruct

    public void jdbcInit(){

        Date curDate = new Date();

        PasswordEncoder encoder = new BCryptPasswordEncoder();

        List<GrantedAuthority> list = new ArrayList<>();
```

```
//参数角色名称，需要以"ROLE_"开头， 后面加上自定义的角色名称

GrantedAuthority authority = new
SimpleGrantedAuthority("ROLE_"+"READ");

list.add(authority);


SysUser user = new SysUser(

    "lisi",encoder.encode("456"),"李四
",true,true,true,true,curDate, curDate, list
);

userMapper.insertSysUser(user);


List<GrantedAuthority> list2 = new ArrayList<>();

GrantedAuthority authority2 = new
SimpleGrantedAuthority("ROLE_"+"ADMIN");

GrantedAuthority authority3 = new
SimpleGrantedAuthority("ROLE_"+"USER");

list.add(authority2);

list.add(authority3);


SysUser user2 = new SysUser(
```

```
        "admin",encoder.encode("admin"),"管理员",true,true,true,true,curDate, curDate, list2);

        userMapper.insertSysUser(user2);

    }
}
```

3.12 mapper 文件

在 resources 目录创建了 mapper, 存放 mybatis 的 xml 文件

1. SysUser 表的操作 mapper 文件

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE mapper

    PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"

    "http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<mapper namespace="com.wkcto.mapper.SysUserMapper">

    <!--定义 列和 属性的对应关系-->

    <resultMap id="userMapper" type="com.wkcto.entity.SysUser">

        <id column="id" property="id"/>

    </resultMap>

</mapper>
```

```
<result column="username" property="username"/>
<result column="password" property="password" />
<result column="realname" property="realname" />
<result column="isenable" property="isEnabled" />
<result column="islock" property="isLocked" />
<result column="iscredentials" property="isCredentials" />
<result column="createtime" property="createTime" />
<result column="logintime" property="loginTime" />
<result column="isexpire" property="isExpired" />
</resultMap>

<insert id="insertSysUser">

    insert into sys_user(username,password,realname,
        isenable,islock,iscredentials,createtime,logintime)

    values(#{username},#{password},#{realname},#{isEnabled},
        #{isLocked},#{isCredentials},#{createTime},#{loginTime})
</insert>

<select id="selectSysUser" resultMap="userMapper">
```

```
select id, username,password,realname,isexpire,
       isenable,islock,iscredentials,createtime,logintime
from sys_user where username=#{username}

</select>

</mapper>
```

2.SysRoleMapper 文件

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE mapper

    PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"

    "http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<mapper namespace="com.wkcto.mapper.SysRoleMapper">

    <!--定义 列和 属性的对应关系-->

    <resultMap id="roleMapper" type="com.wkcto.entity.SysRole">

        <id column="id" property="id"/>

        <result column="rolename" property="name"/>

        <result column="rolememo" property="memo" />

    </resultMap>
```

```
<select id="selectRoleByUser" resultMap="roleMapper">

    select r.id, r.rolename,r.rolmemo from sys_user_role ur , sys_role r

    where ur.roleid = r.id and ur.userid=#{userid}

</select>

</mapper>
```

3.13 index.html

在 resources/static 目录中，创建 index.html

```
<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <title>Title</title>

</head>

<body>

    <p>验证访问</p>

    <a href="/access/user">验证 zs</a> <br/>

    <a href="/access/read">验证 lisi</a><br/>

    <a href="/access/admin">验证 admin</a><br/>
```



```
</body>
```

```
</html>
```

3.14. application.properties

```
#数据源配置
```

```
spring.datasource.driver-class-name=com.mysql.jdbc.Driver
```

```
spring.datasource.url=jdbc:mysql://localhost:3306/ssm
```

```
spring.datasource.username=root
```

```
spring.datasource.password=123456
```

```
#mybatis 需要的 mapper 文件位置
```

```
mybatis.mapper-locations=classpath:/mapper/*Mapper.xml
```

```
# 别名
```

```
mybatis.type-aliases-package=com.wkcto.entity
```

3.14 默认的登录页面

1.访问地址 /login

2.请求方式 post

3.请求参数

用户名 username

密码 password