WebSystique
learn together

Spring Security 4 Secure View Fragments using taglibs

July 28, 2015 websystiqueadmin

This tutorial shows you how to secure view layer, show/hide parts of jsp/view based on logged-in user's roles, using Spring Security tags in Spring MVC web application.

First of all, in order to use Spring Security tags, we need to include **spring-security-taglibs** dependency in pom.xml as shown below:

```
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-taglibs</artifactId>
    <version>4.0.1.RELEASE</version>
</dependency>
```

Then the next step would be to include taglib in your views/JSP's.

```
<%@ taglib prefix="sec" uri="http://www.springframework.org/securit
```

Finally, we can use Spring Security expresssions like hasRole, hasAnyRole, etc.. in Views as shown below:

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1
```

Websystique
199 likes
Liked

WebSyst
Foll
+ 100

Recent Posts

Spring 4 MVC+AngularJS CRUD Application using ngResource

Angularjs Server Communication using ngResource-CRUD Application

AngularJS Custom-Directives controllers, require option guide

AngularJS Custom-Directives transclude, ngTransclude guide

AngularJS Custom-Directives

```
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<%@ taglib prefix="sec" uri="http://www.springframework.org/securit
<html>
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=ISC
    <title>Welcome page</title>
</head>
<body>
    Dear <strong>${user}</strong>, Welcome to Home Page.
    <a href="<c:url value="/logout" />">Logout</a>

    <br/>
    <br/>
    <div>
        <label>View all information| This part is visible to Everyc
    </div>

    <br/>
    <div>
        <sec:authorize access="hasRole('ADMIN')">
            <label><a href="#">Edit this page</a> | This part is vi
        </sec:authorize>
    </div>

    <br/>
    <div>
        <sec:authorize access="hasRole('ADMIN') and hasRole('DBA')"
            <label><a href="#">Start backup</a> | This part is visi
        </sec:authorize>
    </div>
</html>
```

replace option guide

That's all you need to conditionally show/hide view fragments based on roles, using Spring Security expressions in your Views.

Below is the Security Configuration used for this example:

```
package com.websystique.springsecurity.configuration;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.annotation.Configuration;
import org.springframework.security.config.annotation.authenticatic
import org.springframework.security.config.annotation.web.builders.
import org.springframework.security.config.annotation.web.configura
import org.springframework.security.config.annotation.web.configura

@Configuration
@EnableWebSecurity
public class SecurityConfiguration extends WebSecurityConfigurerAda


    @Autowired
    public void configureGlobalSecurity(AuthenticationManagerBuilde
        auth.inMemoryAuthentication().withUser("bill").password("ab
        auth.inMemoryAuthentication().withUser("admin").password("r
        auth.inMemoryAuthentication().withUser("dba").password("roc
    }

    @Override
    protected void configure(HttpSecurity http) throws Exception {

      http.authorizeRequests()
        .antMatchers("/", "/home").access("hasRole('USER') or hasRc
        .and().formLogin().loginPage("/login")
        .usernameParameter("ssoId").passwordParameter("password")
        .and().exceptionHandling().accessDeniedPage("/Access_Deniec
    }
}
```

Above security configuration in XML configuration format would be:

```
<beans:beans xmlns="http://www.springframework.org/schema/security"
    xmlns:beans="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/security http://www.sprir

    <http auto-config="true" >
        <intercept-url pattern="/"     access="hasRole('USER') or h
        <intercept-url pattern="/home" access="hasRole('USER') or h
        <form-login  login-page="/login"
                     username-parameter="ssoId"
                     password-parameter="password"
                     authentication-failure-url="/Access_Denied" />
    </http>

    <authentication-manager >
        <authentication-provider>
            <user-service>
                <user name="bill"  password="abc123"  authorities="
                <user name="admin" password="root123" authorities="
                <user name="dba"   password="root123" authorities="
            </user-service>
        </authentication-provider>
    </authentication-manager>


</beans:beans>
```

And the controller:

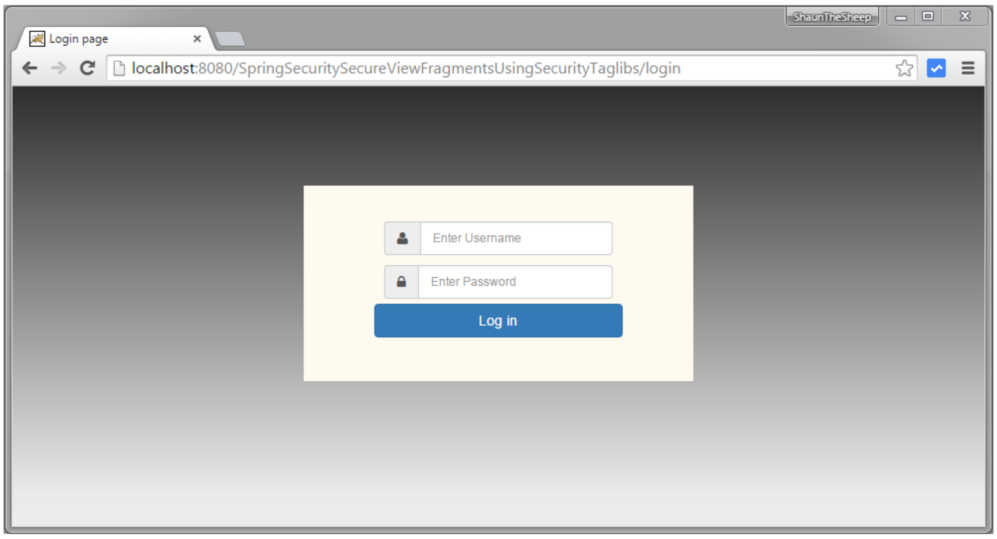
```
package com.websystique.springsecurity.controller;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.springframework.security.core.Authentication;
import org.springframework.security.core.context.SecurityContextHol
import org.springframework.security.core.userdetails.UserDetails;
import org.springframework.security.web.authentication.logout.Secur
import org.springframework.stereotype.Controller;
import org.springframework.ui.ModelMap;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;

@Controller
public class HelloWorldController {


    @RequestMapping(value = { "/", "/home" }, method = RequestMethc
    public String homePage(ModelMap model) {
        model.addAttribute("user", getPrincipal());
        return "welcome";
    }

    @RequestMapping(value = "/Access_Denied", method = RequestMethc
    public String accessDeniedPage(ModelMap model) {
        model.addAttribute("user", getPrincipal());
        return "accessDenied";
    }
    @RequestMapping(value = "/login", method = RequestMethod.GET)
    public String loginPage() {
        return "login";
    }

    @RequestMapping(value="/logout", method = RequestMethod.GET)
    public String logoutPage (HttpServletRequest request, HttpServl
        Authentication auth = SecurityContextHolder.getContext().ge
        if (auth != null){
            new SecurityContextLogoutHandler().logout(request, resp
        }
        return "redirect:/login?logout";
    }

    private String getPrincipal(){
        String userName = null;
        Object principal = SecurityContextHolder.getContext().getAu

        if (principal instanceof UserDetails) {
            userName = ((UserDetails)principal).getUsername();
        } else {
            userName = principal.toString();
        }
        return userName;
    }

}
```

Rest of application code is same as other posts in this series.

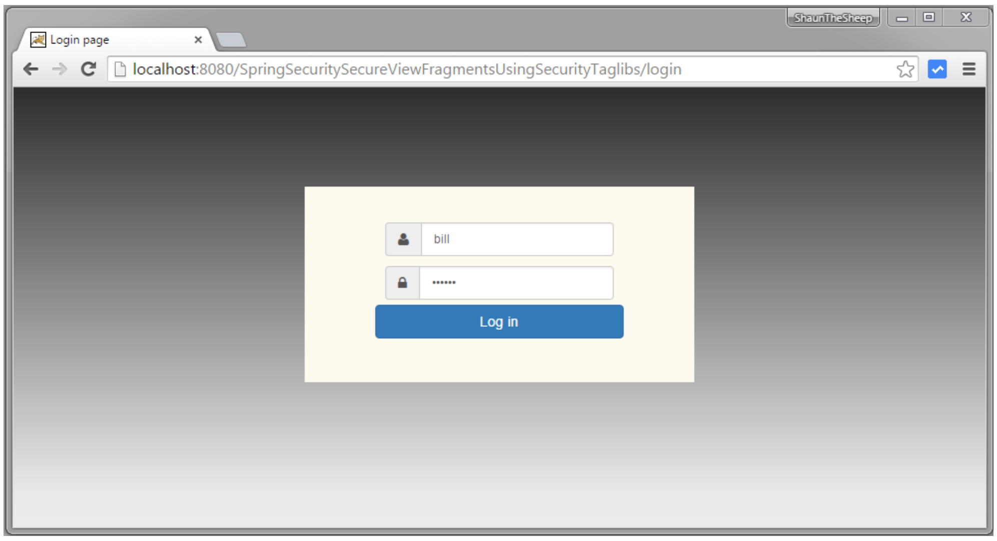
Deploy & Run

Download complete code of this project using download button shown at the bottom of this post. Build and deploy it on Servlet 3.0 container(Tomcat7/8).

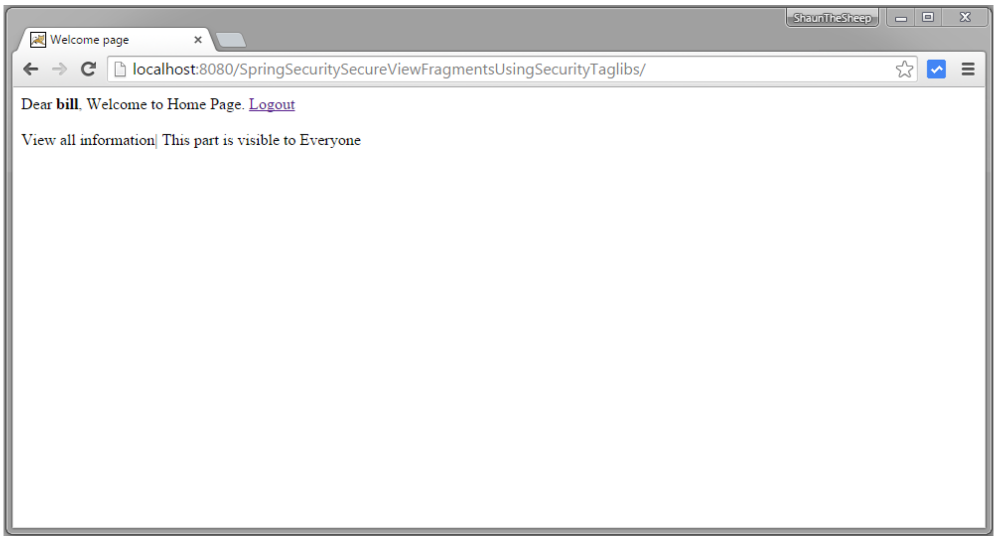
Open browser and access homepage at **localhost:8080/SpringSecuritySecureViewFragmentsUsingSecurityTaglibs/**, you will be prompted for login.


Login page
localhost:8080/SpringSecuritySecureViewFragmentsUsingSecurityTaglibs/login
ShaunTheSheep
Enter Username
Enter Password
Log in

Provide USER credentials.


Login page
localhost:8080/SpringSecuritySecureViewFragmentsUsingSecurityTaglibs/login
bill
••••••
Log in

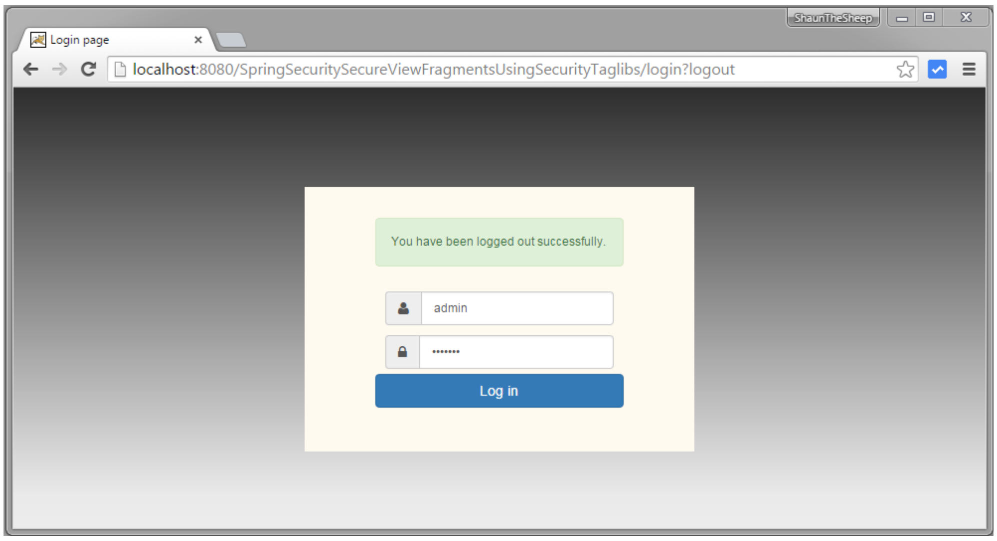
You can see that limited information is shown on page.


Welcome page
localhost:8080/SpringSecuritySecureViewFragmentsUsingSecurityTaglibs/

Dear **bill**, Welcome to Home Page. Logout

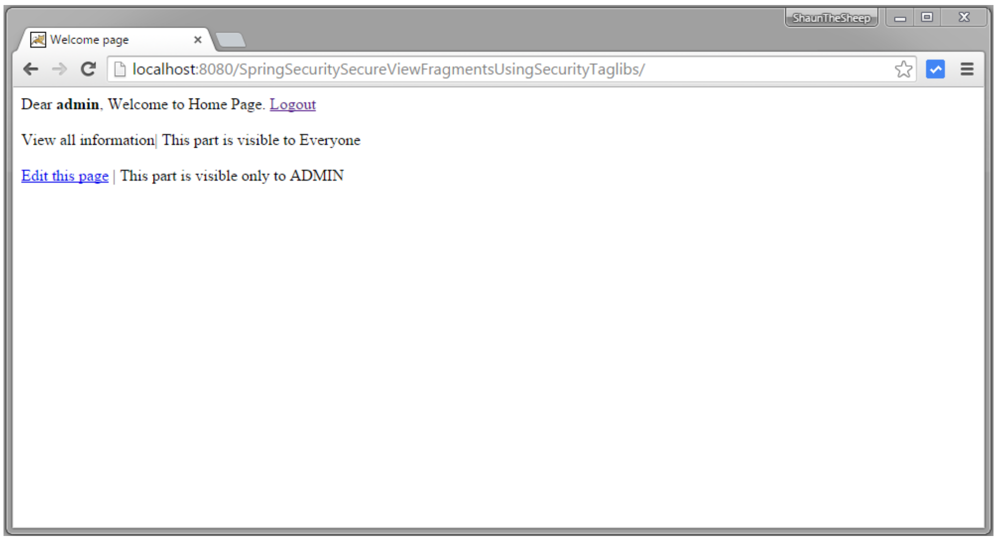
View all information| This part is visible to Everyone

Now click on logout and login with ADMIN role


Login page
localhost:8080/SpringSecuritySecureViewFragmentsUsingSecurityTaglibs/login?logout
admin
•••••••
Log in

You have been logged out successfully.

Submit, you will see that operation related to ADMIN role is accessible.



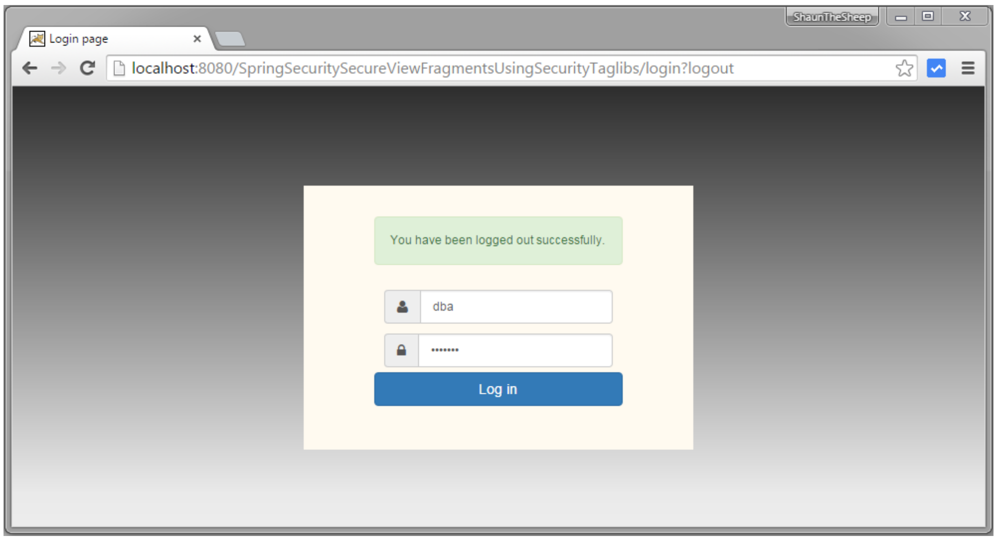
localhost:8080/SpringSecuritySecureViewFragmentsUsingSecurityTaglibs/

Dear **admin**, Welcome to Home Page. Logout

View all information| This part is visible to Everyone

Edit this page | This part is visible only to ADMIN

Now logout, and login with DBA role.



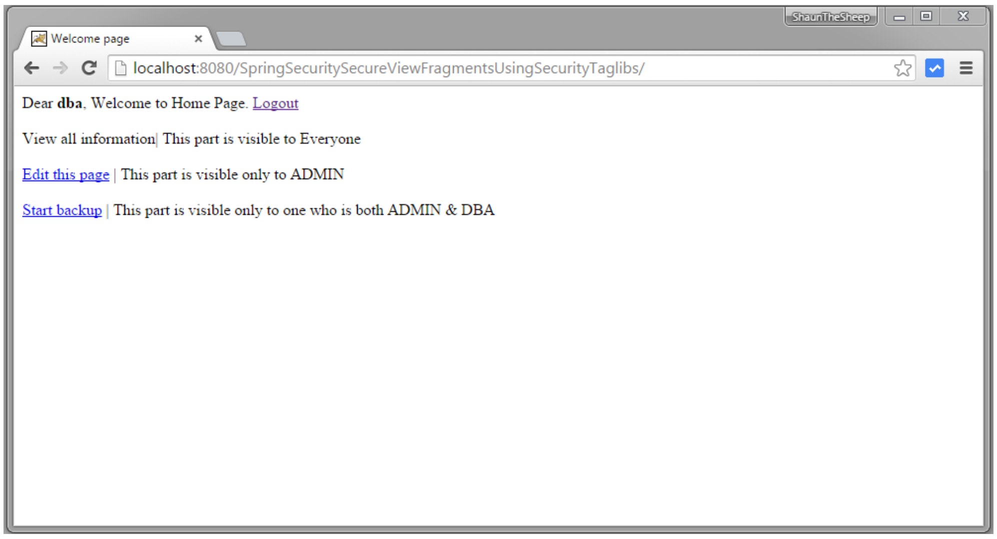
localhost:8080/SpringSecuritySecureViewFragmentsUsingSecurityTaglibs/login?logout

You have been logged out successfully.

dba

•••••••

Submit, you will see that operation related to DBA role is accessible.



localhost:8080/SpringSecuritySecureViewFragmentsUsingSecurityTaglibs/

Dear **dba**, Welcome to Home Page. Logout

View all information| This part is visible to Everyone

Edit this page | This part is visible only to ADMIN

Start backup | This part is visible only to one who is both ADMIN & DBA

That's it. Next post shows you how to use role based login. That means redirecting users to different URLs upon login according to their assigned roles.

Download Source Code

Download Now!

References

- Spring Security Expressions
- Spring Security 4 Project Page
- Spring Security 4 Reference Manual

websystiqueadmin

If you like tutorials on this site, why not take a step further and connect me on Facebook , Google Plus & Twitter as well? I would love to hear your thoughts on these articles, it will help me improve further our learning process.

If you appreciate the effort I have put in this learning site, help me improve the visibility of this site towards global audience by sharing and linking this site from within and beyond your network. You & your friends can always link my site from your site on www.websystique.com, and share the learning.

After all, we are here to learn together, aren't we?

Related Posts:

1. **Spring Security 4 Logout Example**
2. **Spring Security 4 Hibernate Role Based Login Example**
3. **Spring Security 4 Method security using @PreAuthorize,@PostAuthorize, @Secured, EL**
4. **Spring Security 4 Hello World Annotation+XML Example**

spring-security. permalink.

← Spring Security 4 Logout Example

Spring Security 4 Hibernate Integration Annotation+XML Example →

0 Comments websystique

1 Login

Recommend Share Sort by Best

Start the discussion…

Be the first to comment.

ALSO ON **WEBSYSTIQUE** WHAT'S THIS?

Spring Security 4 Hibernate Integration Annotation+XML

31 comments • 6 months ago

websystique — Hi Marcin, Look at this post, here we are using a simple cretiria to fetch all

Jackson Json Annotations Example

2 comments • 7 months ago

websystique — Glad you liked it.

Spring Security 4 Method security using

2 comments • 6 months ago

websystique — Hi Nihel, For that, you just need to change authenitcation method, like

Spring MVC 4 File Download Example

4 comments • 6 months ago

websystique — You don't have to.You can use either of tomcat 7/8

Copyright © 2014-2015 WebSystique.com. All rights reserved.