

Spring Security 5

What is Spring Security?

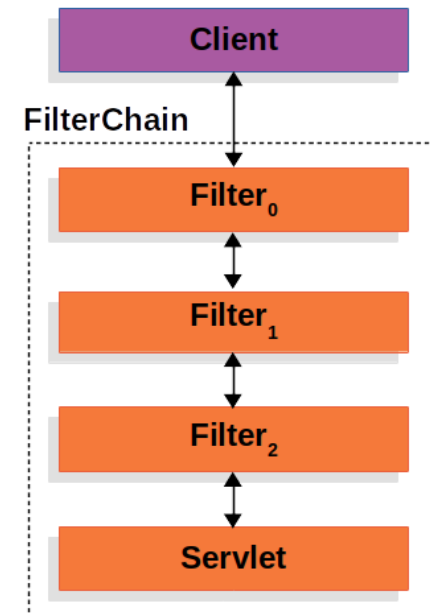
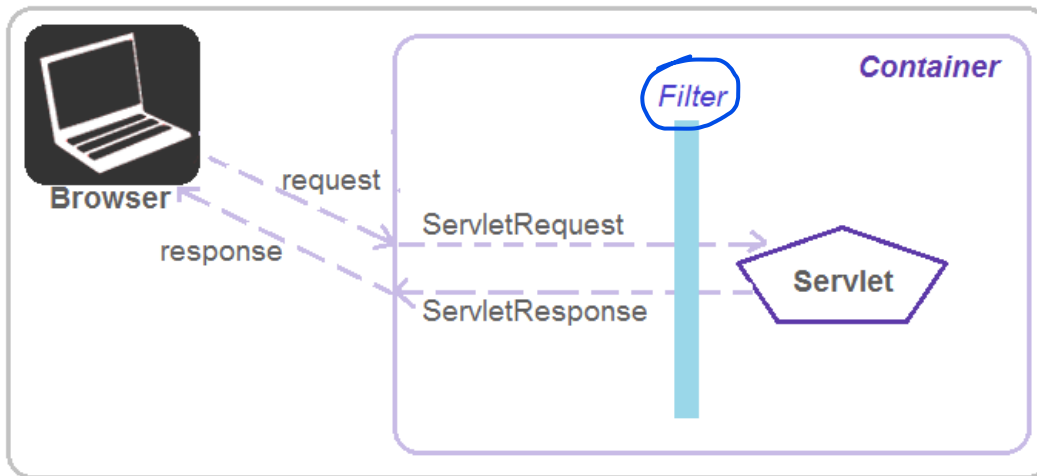
강력하고 맞춤형인 인증과 접근 관련 프레임워크다.

- Powerful and highly customizable authentication and authorization framework
- De-facto standard for securing Spring-based applications

스프링 기반 애플리케이션 보안의 표준이라 볼 수 있음

Spring Security

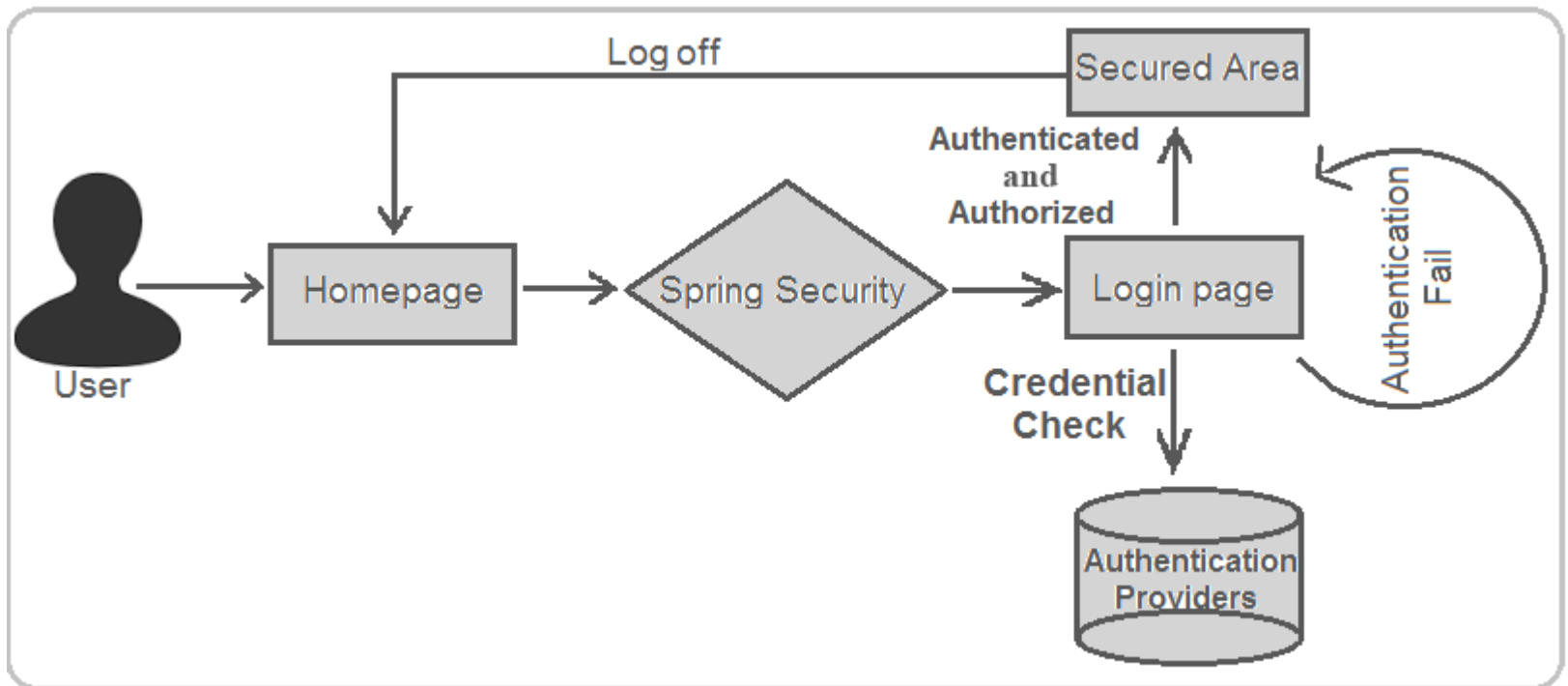
- Spring Security is entirely based on servlet filter
필터에 기반을 두고있다
 - A **Filter** intercepts the requests and responses (to and from Servlet) and can pre-process and post-process



Spring Security

(Use case)

여기 기준



4p 선행

Spring Security

(Use case)

1. The user reaches the application or homepage of the application and clicks on a secure link
2. The moment the user clicks on the secured link, Spring Security brings the login page
3. The login page will perform a credential check from the authentication provider
4. An authentication failure happens if wrong credentials are given by the user; otherwise, the user will be allowed to the secured area
5. When the user clicks on logout, they will be directed to the homepage

1. Simple Spring Security WebApp

예시 코드

```
HomeController.java ✖  
package kr.ac.hansung;
```

```
import org.springframework.stereotype.Controller;
```

```
/**  
 * Handles requests for the application home page.  
 */
```

```
@Controller
```

```
public class HomeController {
```

```
    @RequestMapping("/public")
```

```
    public String accessPublicPage(Model model) {
```

```
        model.addAttribute("message", "This page is publicly accessible. No authentication is required to view.");
```

```
        return "public";
```

```
    }
```

```
    @RequestMapping("/secured/mypage")
```

```
    public String accessSecuredPage(Model model) {
```

```
        model.addAttribute("message", "Only you are authenticated and authorized to view this page.");
```

```
        return "secured/mypage";
```

```
    }
```

```
}
```

① 예는 누구나 접근이 가능하고,

② 예는 허가받은 사람만

접근 가능하게 하려면 어떻게 할까?

①

②

Simple Spring Security WebApp

public.jsp ☒

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
```

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/strict.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>Insert title here</title>

</head>

<body>

<html>

<title>Public Page</title>

<body>

<h4>\${message}</h4>

</body>

</html>

</body>

</html>

mypage.jsp ☒

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
```

<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/strict.dtd">

<html>

<head>

<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">

<title>Insert title here</title>

</head>

<body>

<html>

<title>My Secured Page</title>

<body>

<h2>Hello World!</h2>

<h4>\${message}</h4>

</body>

</html>

</body>

</html>

What if you want to show
this secured page
to authorized people only?

허가받은 사람만 보이면?

1) Spring Security Maven Dependencies

Add Spring Security Dependencies in pom.xml

POM.xml

라이브러리 설치

```
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-core</artifactId>
    <version>${spring-security.version}</version>
</dependency>
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-web</artifactId>
    <version>${spring-security.version}</version>
</dependency>
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-config</artifactId>
    <version>${spring-security.version}</version>
</dependency>
```


2) Enabling Spring Security

- Spring Security uses DelegatingFilterProxy, which will be configured to intercept every request

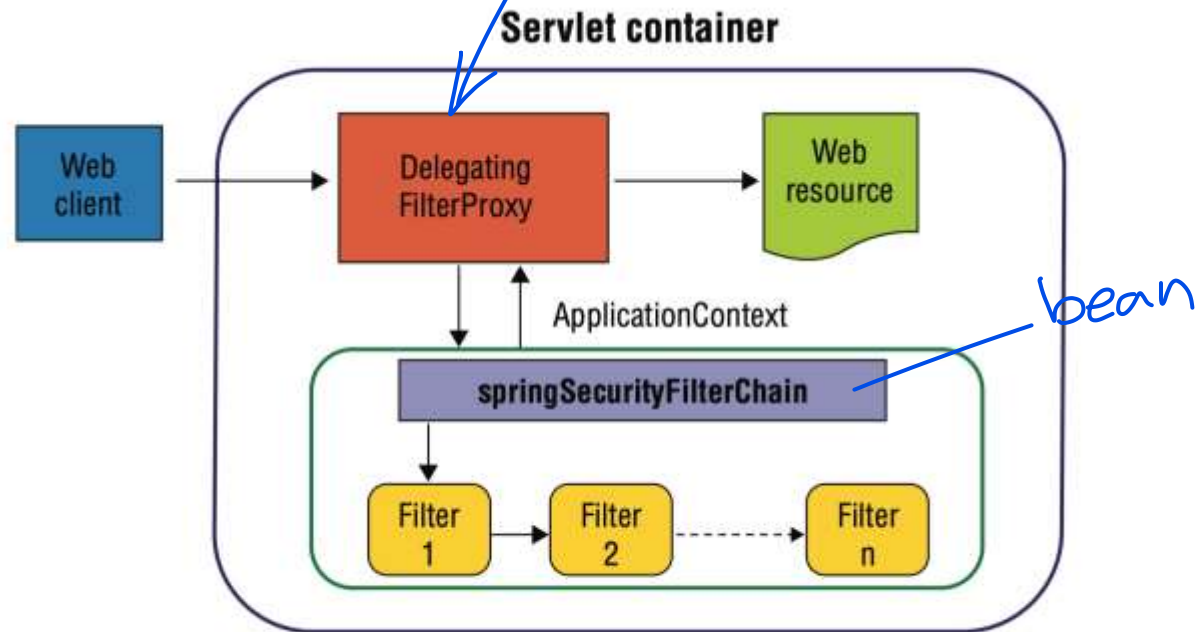


FIGURE 12-2

Enabling Spring Security

web.xml

활성화하기

<filter>

변경 금지

<filter-name>springSecurityFilterChain</filter-name>

<filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>

</filter>

<filter-mapping>

<filter-name>springSecurityFilterChain</filter-name>

<url-pattern>/*</url-pattern>

</filter-mapping>

root 아래의 모든 요청을 받는다

3) Spring Security Configuration

security-context.xml

새로운 설정 파일

```
<?xml version="1.0" encoding="UTF-8"?>
<b:beans xmlns="http://www.springframework.org/schema/security"
  xmlns:b="http://www.springframework.org/schema/beans"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://www.springframework.org/schema/beans
    https://www.springframework.org/schema/beans/spring-beans.xsd
    http://www.springframework.org/schema/security
    https://www.springframework.org/schema/security/spring-security.xsd">
```

User authentication
with in-memory
definition

Authentication

```
<authentication-manager>
  <authentication-provider>
    <user-service>
      <user name="nykim" authorities="ROLE_USER" password="{noop}letmein" />
    </user-service>
  </authentication-provider>
</authentication-manager>
```

(Spring Security 5) General format for a password
{id}encodedPassword "{bcrypt}\$2a\$10\$q..."

letmein의 암호화
Authorization

권한

```
<http auto-config="true" use-expressions="true">
  <intercept-url pattern="/public" access="permitAll" />
  <intercept-url pattern="/secured/**" access="hasRole('ROLE_USER')" />
</http>
```

```
</b:beans>
```

Spring Security Configuration

- `<authentication-manager>` handles authentication of requests and uses the mechanism provided by `<authentication-provider>` to authenticate an user
 - To make the example simple, I have defined one hardcoded user with username as "nykim", password as "letmein" and *authorities* as "ROLE_USER"
 - *Authorities* can take comma separated list of roles assigned to the particular user
 - `<intercept-url>` defines a *pattern* for request URLs which need to be secured
 - Attribute *access* defines roles of a user who is authorized to see requested URLs matching with that pattern
 - *auto-config='true'* automatically enables form based login, basic authentication and logout mechanism
- 로그인이나 권한을 자동 처리 해줌

Spring Security Configuration

Do not forget to put security configuration file
(security-context.xml)
in contextConfigLocation setting

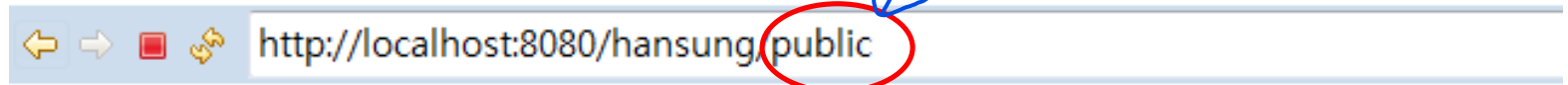
web.xml

```
<context-param>
  <param-name>contextConfigLocation</param-name>
  <param-value>
    /WEB-INF/spring/root-context.xml
    /WEB-INF/spring/appServlet/security-context.xml ← 37v
  </param-value>
</context-param>

<listener>
  <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
</listener>
```

4) Demo

Public Page



This page is publicly accessible. No authentication is required to view.

Demo

Secured/Protected Page

① <http://localhost:8080/hansung/secured/mypage>

We have been intercepted by DelegatingFilterProxy and redirected to spring defined login form at [http:// ... /login](http://.../login)

② <http://localhost:8080/hansung/login>

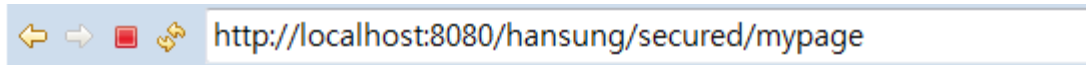
login 으로
리다이렉트 해버림

Please sign in

default login form
provided by spring
스프링 제공 form

Demo

My Secured Page
(after login)



Hello World!

Only you are authenticated and authorized to view this page.

Spring security will redirect us to the initially requested URL and we'll be able to view the content of the secured page

2. Expression-Based Access Control

접근 권한 명시

- Use Spring Expression Language(SpEL) expressions as an authorization mechanism

Expression	Description
hasRole([role])	Returns true if the current principal has the specified role.
hasAnyRole([role1,role2]) 아	Returns true if the current principal has any of the supplied roles (given as a comma-separated list of strings)
principal	Allows direct access to the principal object representing the current user

```
<security:http auto-config="true" use-expressions="true">  
  <security:intercept-url pattern="/secured/**"  
    access="hasRole('ROLE_USER')" />  
</security:http>
```

ROLE_USER 권한 필요

Expression-Based Access Control

Expression	Description
authentication	Allows direct access to the current Authentication object obtained from the SecurityContext
permitAll	Always evaluates to true
denyAll	Always evaluates to false
isAnonymous()	Returns true if the current principal is an anonymous user
isRememberMe()	Returns true if the current principal is a remember-me user
isAuthenticated()	Returns true if the user is not anonymous
isFullyAuthenticated()	Returns true if the user is not an anonymous or a remember-me user

```
<security:http auto-config="true" use-expressions="true">  
  <security:intercept-url pattern="/secured/**"  
    access="isAuthenticated()" />  
</security:http>
```

인증 필요

3. Other Authentication Provider (Using Database)

데이터 베이스 연동

- Use jdbc-user-service to define a query to perform database authentication

security-context.xml

```
<security:authentication-manager>
  <security:authentication-provider>
    <security:user-service>
      <security:user name="nykim" authorities="ROLE_USER"
        password="letmein" />
    </security:user-service>
  </security:authentication-provider>

  <security:authentication-provider>
    <security:jdbc-user-service data-source-ref="dataSource"
      <u>users</u>-by-username-query=
        "select username, password, enabled from users where username = ?"
      <u>authorities</u>-by-username-query=
        "select username, authority from authorities where username = ?" />
    </security:authentication-provider>
  </security:authentication-manager>
```

비밀번호

password = "{noop}letmein"
(Spring Security 5)

DB

SQL문

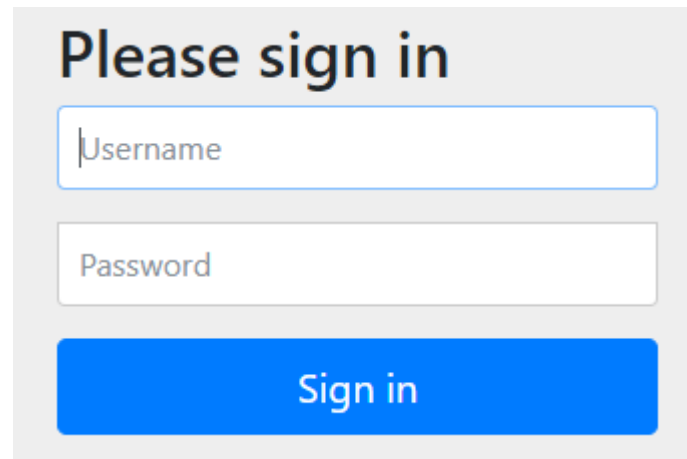
users	
username	VARCHAR(45)
password	VARCHAR(80)
enabled	FIXED
Indexes	

authorities	
username	VARCHAR(45)
authority	VARCHAR(45)
Indexes	

4. Creating a Custom Login Form

- Default Login Form

<http://localhost:8080/csmall/login>

A screenshot of the default Spring Security login form. It features a light gray background with the text "Please sign in" at the top. Below this are two input fields: the first is labeled "Username" and the second is labeled "Password". At the bottom of the form is a blue button with the text "Sign in" in white.

Spring Security will create a default URL at *'/login'*
and render a default login form

Creating a Custom Login Form (security-context.xml)

- If you want to supply your own login page, you could use `<form-login>` element

```
<security:http auto-config="true" use-expressions="true">
  <security:intercept-url pattern="/admin/**"
    access="hasRole('ROLE_ADMIN')" />
  <security:form-login login-page="/login" />
</security:http>
```

↑
이것만 추가하면 로그인 창 수정가능

The 'form-login' element just overrides the default settings

Creating a Custom Login Form (security-context.xml)

The line `login-page="/Login"` instructs Spring Security

- ① when authentication is required, redirect the browser to **/login** 인증이 필요하면 리다이렉트
- we are in charge of rendering the login page when **/login** is requested
- ② when authentication attempt fails, redirect the browser to **/login?error** (since we have not specified otherwise)
- we are in charge of rendering a failure page when **/login?error** is requested
인증 실패시 error 파라미터 전달, 로그인 실패 페이지 전송

Creating a Custom Login Form

security:form-login attribute

Attribute	Default value	Note
login-page	/login	Should render a web page
username-parameter	username	
password-parameter	password	
login-processing-url	/login , <u>POST</u>	
authentication-failure-url	<u>/login?error</u>	<u>Should render a web page</u>

the URL used to process the login request by spring

로그인 처리 시 쓸 URL 이 필요,

POST 로 보내면 스프링이 알아서 해줌

1) Custom Login Form

LoginController.java

```
@Controller
public class LoginController {

    @RequestMapping(value="/login", method = RequestMethod.GET)
    public String login() {

        return "login";
                view
    }
}
```


Custom Login Form

login.jsp

login processing module
of spring security

```
<form action= "<c:url value="/login" />" method='POST'>
```

```
<table>
```

```
<tr><td>User:</td><td><input type='text' name='username' value=""></td></tr>
```

```
<tr><td>Password:</td><td><input type='password' name='password'></td></tr>
```

```
<tr><td colspan='2'><input name="submit" type="submit" value="Login" /></td></tr>
```

```
<input type="hidden" name="${csrf.parameterName}" value="${csrf.token}"/>
```

```
</table>
```

```
</form>
```

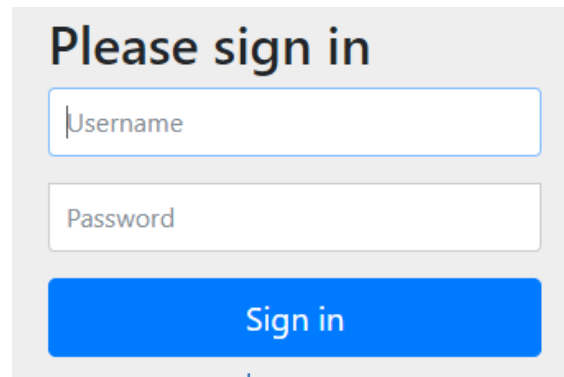
사용자에게
안보임

CSRF protection
공격

공격 방지용
인증 토큰

Custom Login Form

Login Page



Please sign in

Username

Password

Sign in



- URL: /login
 - Method: post
- > Spring performs login process by accessing database

2) Authentication Failure

인증 실패의 경우

if you try to login with *invalid* username and password, you will be redirected to same login page (**authentication-failure-url** --> **'/login?error'**) and authentication failure message will also be displayed in the login screen.

LoginController.java

@Controller

```
public class LoginController {
```

```
    @RequestMapping(value="/login", method = RequestMethod.GET))
```

```
    public String login(@RequestParam(value="error", required=false)  
                        String error, Model model) {
```

```
        if(error != null) {  
            model.addAttribute("errorMsg", "Invalid username and password");  
        }
```

```
        return "login";
```

```
    }
```

↑
이러한 메시지 표시

Authentication Failure

login.jsp

```
<form action="<c:url value="/login" />" method='POST'>
```

For failed
user authentication

```
<c:if test="${not empty errorMsg}"> ← 이걸 표시하기 위한  
  <div style="color: #ff0000;"> <h3> ${errorMsg} </h3> </div>  
</c:if> 이걸을 출력
```

```
<table>
```

```
<tr><td>User:</td><td><input type='text' name='username' value=''></td></tr>
```

```
<tr><td>Password:</td><td><input type='password' name='password' /></td></tr>
```

```
<tr><td colspan='2'><input name='submit' type='submit' value='Login' /></td></tr>
```

```
<input type='hidden' name='${_csrf.parameterName}' value='${_csrf.token}' />  
</table>
```

```
</form>
```

Authentication Failure

Login Error

localhost:8080/helloSpringMVC/login?error

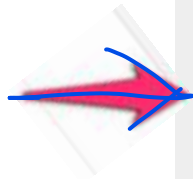
Please sign in

Invalid username and
password

Username

Password

Sign in



3) Adding Log out

로그아웃

security-context.xml

```
<http auto-config="true" use-expressions="true">
  <intercept-url pattern="/" access="permitAll" />

  <form-login login-page="/login" />

  <logout />
</http>
```

logout attribute

Attribute	Default value	Note
logout-url	/logout, POST	
logout-success-url 로그아웃 성공	/login? <u>logout</u>	Should render a web page

Log out processing module by spring

Adding Log out

home.jsp

Solution 1:

Show current offers

Add a new offer

Logout

링크로 하는 법

`<c:if test="${pageContext.request.userPrincipal.name != null}">` ← 로그인 했으면
`Logout` 보여주기
`</c:if>` 클릭 시 logout 이라는 id 가진 것 Submit
`<form id="logout" action="<c:url value="/logout" />"method="post">`
`<input type="hidden" name="${_csrf.parameterName}"value="${_csrf.token}" />`
`</form>`

Adding Log out

home.jsp

Solution 2:

[Show current Offers](#)

[Add a new offer](#)

Log out

버튼으로 하는 방법

```
<c:if test= "${pageContext.request.userPrincipal.name != null}">
```

```
  <c:url var= "logoutUrl" value= "/logout"/>
```

```
  <form class= "form-inline" action= "${logoutUrl}" method= "post">
```

```
    <input type= "submit" value= "Log out" />
```

```
    <input type= "hidden" name= "${_csrf.parameterName}" value= "${_csrf.token}"/>
```

```
  </form>
```

```
</c:if>
```


Adding Log out

LoginController.java

```
@RequestMapping(value="/login", method = RequestMethod.GET)
public String login(
    @RequestParam(value = "error", required = false) String error,
    @RequestParam(value="logout", required=false) String logout,
    Model model) {

    if (error != null) {
        model.addAttribute("errorMsg", "Invalid username and password");
    }

    if(logout != null) { error 동일한 원리
        model.addAttribute("logoutMsg", "You have been logged out successfully ");
    }

    return "login";
}
```

Adding Log out

login.jsp

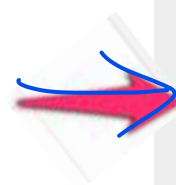
<h3>Custom Login with Username and Password</h3>

```
<c:if test= "${not empty logoutMsg}" > 에러와 같은 원인  
  <div style="color: #0000ff;" > <h3> ${logoutMsg} </h3> </div>  
</c:if>
```

```
<form action= "<c:url value= "/login" />" method= 'POST'>
```

...

```
</form>
```



localhost:8080/helloSpringMVC/login?logout

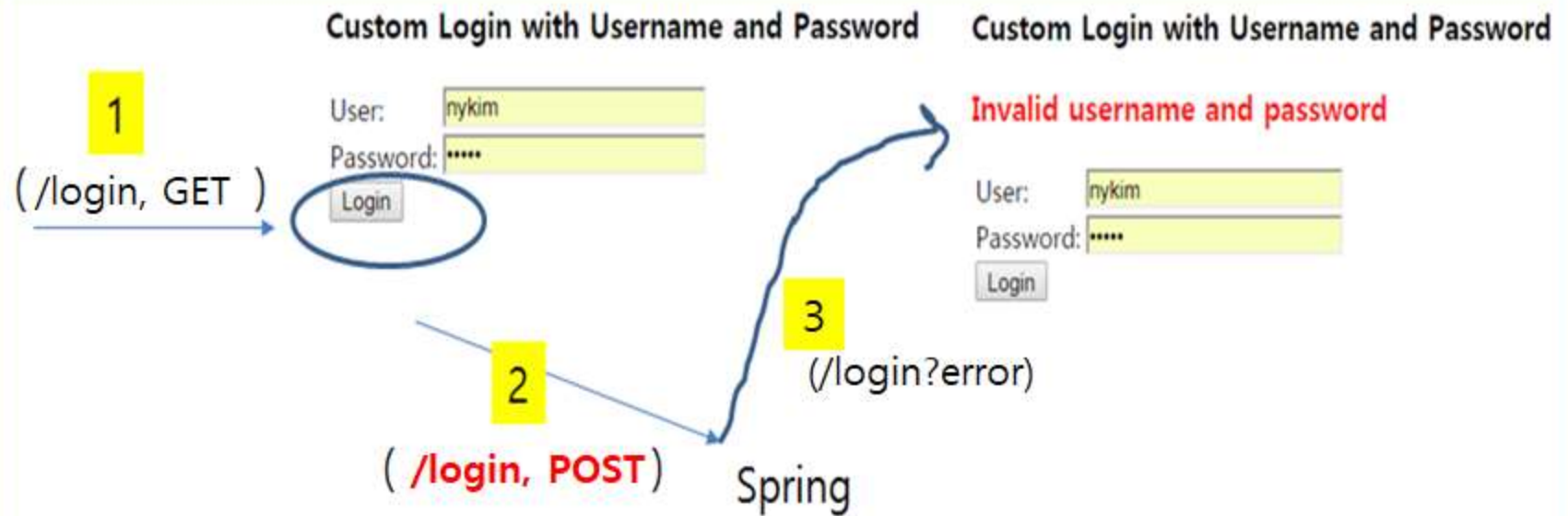
Please sign in

You have been logged out
successfully

Username

Password

Sign in



2,4 는 스프링이 해줌

[Show current offers](#)

[Add a new offer](#)

[Logout](#)

4 (/logout, POST)

Spring

5 (/login ?logout, GET)

You have been logged out successfully

User: nykim
Password:
Login