

1 Spring Security  
2 1. 소개  
3 1)웹 보안 개요  
4 -All-in-One Java 애플리케이션 개발, 전병선, 2014 참조  
5 -웹 어플리케이션을 작성하면서 보안과 관련해서 고려할 사항은 다음과 같다.  
6 --인증(authentication) : 사용자가 자신임을 증명하는 속성  
7 --권한(authorization) : 사용자가 어떤 리소스에 접근하게 할지를 결정하는 속성  
8 --기밀성(privacy or confidentiality) : 권한이 없는 사용자에게 접근을 허용하지 않게 하는 속성  
9 --무결성(integrity) : 권한이 없는 사용자가 데이터를 변경하기 못하게 하는 속성  
10 --부인방지(repudiation) : 사용자가 자신이 했다는 것을 부인하지 못하게 하는 속성  
11  
12 2)Spring Security는 이들 보안 요소 중에서 인증과 권한 서비스를 제공한다.  
13 3)인증은 사용자가 자신임을 증명하는 것이고, 권한은 사용자가 어떤 리소스에 접근할 수 있는지를 결정하는 것이다.  
14 4)우리가 웹 어플리케이션의 리소스에 접근할 수 있는지 여부는 권한과 관련되어 있다.  
15 5)사용자가 권한이 있는지를 알기 위해서는 권한을 가진 사용자인지 인증되어야 한다.  
16 6)그러니까 먼저 사용자가 인증되어야 하고, 인증된 사용자에게 부여된 권한으로 리소스에 접근할 수 있는지를 체크한다.  
17 7)사용자를 인증하는 다양한 방법이 있지만 가장 보편적으로 사용하는 방법은 폼 인증 방식이다.  
18 8)사용자가 인증된 후에는 리소스에 접근할 수 있는 권한을 가졌는지를 체크해야 한다.  
19 9)사용자의 권한을 수립하는 첫 번째 단계는 principle 형식으로 사용자를 표현하는 것이다.  
20 10)principle은 리소스에 접근하기 위한 보안 식별자가 할당된 계정 보유자로서, 사용자, 그룹, 서비스, 컴퓨터 등이 principle이 될 수 있다.  
21 11)개별 사용자에게 일일이 권한을 가졌는지를 검사하는 것을 비효율적일 뿐만 아니라 경우에 따라서는 불가능하기도 하다.  
22 12)따라서, 사용자에게 권한을 부여할 때 가장 많이 사용하는 일반적인 방법은 사용자를 역할(role)에 할당하는 역할 기반 방식이다.  
23 13)ROLE\_ADMIN, ROLE\_USER, ROLE\_MNAGER 등의 역할을 정의하고, 개별 사용자에게 역할을 할당한다.  
24  
25 2. 환경설정  
26 1)Spring Legacy Project 생성  
27 -In Package Explorer > right-click > New > Spring Legacy Project  
28 -Project name : SpringSecurityDemo  
29 -Select Spring MVC Project > Next  
30 -com.javasoft.biz > Finish  
31  
32 2)server.xml에 project 추가  
33 -Tomcat v8.5 Server at localhost > right-click > Add and Remove  
34 -SpringSecurityDemo > Add > Finish > OK  
35  
36 3)Spring security library download and install  
37 -In pom.xml 아래와 같이 수정  
38 <java-version>1.8</java-version>  
39 <org.springframework-version>4.3.17.RELEASE</org.springframework-version>  
40 <org.aspectj-version>1.9.1</org.aspectj-version>  
41 <org.slf4j-version>1.6.6</org.slf4j-version>  
42  
43 -In mvnrepository, search 'spring security'  
44 -Select 'Spring Security Core' 4.2.6  
45 -Select 'Spring Security Web' 4.2.6  
46 -Select 'Spring Security Config' 4.2.6  
47 -Select 'Spring Security Taglibs' 4.2.6  
48 -pom.xml > right-click > Run As > Maven clean and Maven install  
49  
50 4)project > right-click > Run As > Run on Server  
51  
52 5)보안 관련 설정 파일 만들기  
53 -/src/main/webapp/WEB-INF/conig folder 생성  
54 -project > right-click > Build Path > Config Build Path

55 -Source tab > Add Folder > check src/main/webapp/WEB-INF/config > OK > Apply  
and Close  
56 -src/main/webapp/WEB-INF/config > right-click > New > Spring Bean Configuration  
File  
57 -File name : securityContext.xml > Next  
58 -Check beans, security > Finish  
59 -In /WEB-INF/web.xml에 보안 관련 설정 파일 등록

```
60     <context-param>
61         <param-name>contextConfigLocation</param-name>
62         <param-value>classpath:securityContext.xml</param-value>
63     </context-param>
```

66 -In securityContext.xml 코드 추가

```
67     <security:http auto-config="true">
68         <!--auto-config='true'를 설정한 것만으로 기본 로그인페이지/HTTP 기본인증/로그아웃
        기능등을 제공한다. -->
69         <!--만일 여기서 use-expressions='true'라고 하면 SpEL을 사용한다는 의미이다. -->
70         <!-- use-expressions은 기본값이 false이다. 이럴때에는 SpEL을 사용하지 않는다는
        의미이다. -->
71         <security:intercept-url pattern="/login.html"
72             access="hasRole('ROLE_USER')" />
73         <security:intercept-url
74             pattern="/welcome.html" access="hasRole('ROLE_ADMIN')" />
75     </security:http>
76
77     --<intercept-url pattern="..." access="ROLE_ANONYMOUS" />
78     --<intercept-url pattern="..." access="IS_AUTHENTICATED_ANONYMOUSLY" />
79     --<intercept-url pattern="..." access="ROLE_USER" />
80     --<intercept-url pattern="..." access="ROLE_ADMIN" />
81     --해당 URL 에 접근하기위한 권한을 설정하여준다.
82     --접근가능한 IP 등을 설정할 수도 있다.
83     --그리고 권한은 위쪽이 우선시된다.
84     --<intercept-url pattern="/login" access="permitAll" />
85     ---/login 으로는 모두 허용해준다./login 을 막아놓으면 안되니까.
86     --<intercept-url pattern="/resources/**" access="permitAll" />
87     ---리소스도 허용
88     --<intercept-url pattern="/**" access="hasRole('ROLE_USER')" />
89     ---나머지는 모두 ROLE_USER 권한을 가진사람만 허용해준다.
```

```
91     <security:authentication-manager>
92         <security:authentication-provider>
93             <security:user-service>
94                 <security:user name="javaexpert" password="12345678"
95                     authorities="ROLE_USER"/>
96                 <security:user name="admin" password="12345678"
97                     authorities="ROLE_ADMIN,ROLE_USER"/>
98             </security:user-service>
99         </security:authentication-provider>
100     </security:authentication-manager>
```

```
100     ---<authentication-manager>          ---인증처리를 위한 최상위 태그
101         <authentication-provider user-service-ref="memberService"/>
102         ----사용자이름/비밀번호를 제공해줄 서비스 등록
103     </authentication-manager>
104     ---위의 예제는 Inmemory로 처리하고 있음.
```

106 -In web.xml 에 코드 추가

107 ---반드시 filter의 이름은 springSecurityFilterChain 이어야 한다.

```

108
109     <filter>
110         <filter-name>springSecurityFilterChain</filter-name>
111         <filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>
112     </filter>
113
114     <filter-mapping>
115         <filter-name>springSecurityFilterChain</filter-name>
116         <url-pattern>/*</url-pattern>
117     </filter-mapping>
118 
```

-In web.xml 에서 코드 수정

```

120
121     <servlet-mapping>
122         <servlet-name>appServlet</servlet-name>
123         <url-pattern>*.html</url-pattern>  <--여기 수정할 것
124     </servlet-mapping>
125 
```

## 6) Test

-/WEB-INF/views/security folder 생성  
 -/WEB-INF/views/security/login.jsp

```

129
130     <body>
131         <h1>login.jsp</h1>
132     </body>
133 
```

-/WEB-INF/views/security/welcome.jsp

```

135
136     <body>
137         <h1>welcome.jsp</h1>
138     </body>
139 
```

-/WEB-INF/views/home.jsp

```

140
141
142     <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
143     <%@ page session="false" %>
144     <html>
145     <head>
146         <title>Home</title>
147     </head>
148     <body>
149     <h1>
150         ${greeting}
151     </h1>
152
153     </body>
154     </html>
155 
```

-HomeController.java 코드 수정

```

156
157
158     @RequestMapping(value = "/index.html", method = RequestMethod.GET)
159     public String home(Model model) {
160         model.addAttribute("greeting", "Hello! Spring Security");
161         return "home";
162     }
163
164     @RequestMapping("/login.html")
165     public String login(Model model) {
166         return "security/login";
167     }

```

```

168
169     @RequestMapping("/welcome.html")
170     public String welcome(Model model) {
171         return "security/welcome";
172     }
173

```

174 -project > right-click > Run As > Run on Server

175  
176 -/login.html로 들어가면 정상적으로 login.jsp 로 이동한다.

177 -username과 password를 미리 저장된 값을 넣지 않으면 Bad credentials로 로그인에러 발생한다.

178 -/welcome.html로 들어가면 User와 Password를 입력할 수 있는 창으로 이동한다.

179 -여기서 user와 password를 잘못 입력하면 'Bad credentials' 에러 메시지가 나온다.

180 -User에 미리 설정한 javaexpert, Password에 12345678을 입력하면 403 Forbidden 오류가 나온다.

181 -즉 권한이 없다는 뜻이다.

182 -만일 미리 설정한 admin/12345678을 입력하면 정상적으로 welcome.jsp로 이동한다.

183 -왜냐하면 이 계정은 ROLE\_ADMIN 권한을 갖고 있기 때문이다.

### 186 3. Login page 만들기

#### 187 1)securityContext.xml 파일 코드 추가

```

188
189 <security:http auto-config="true">
190     <security:form-login login-page="/loginForm.html"
191         username-parameter="j_username"
192         password-parameter="j_password"
193         login-processing-url="/j_spring_security_check" />
194     <security:intercept-url pattern="/login.html"
195         access="hasRole('ROLE_USER')" />
196     <security:intercept-url
197         pattern="/welcome.html" access="hasRole('ROLE_ADMIN')" />
198     <security:csrf disabled="true"/>
199 </security:http>
200

```

201 -form 기반 인증을 사용하고자 할 때는 아래와 같이 설정한다.

202 <form-login login-page="/login" -----사용자가 만든 로그인페이지를 스프링에게 알려준다. 이거 설정안하면 스프링이 만들어준것을 사용.

203 default-target-url="/monitoring" -----로그인성공하면 이동할 페이지설정

204 username-parameter="username"

205 password-parameter="password"

206 authentication-failure-url="/login?error" -----실패시 호출해줄 URL (login 페이지에 error 파라미터를 보내준다)

207 always-use-default-target='true' 이거 안하면 로그인 성공해도 /monitoring 로 제대로 안갈 수 있다.

208 />

209 <logout invalidate-session="true" logout-url="/logout"

logout-success-url="/login?logout" />

210 --로그아웃되면 세션을 초기화한다.

211 --logout-success-url="/login?logout" ----- 로그아웃되면 이동할 페이지

212 --logout-url="/logout" -----로그아웃을 위한 URL설정. 이거 안해주면 디폴트로 j\_spring\_security\_logout 해주면됨.

213 <csrf/> ----- 간단한 설정으로 csrf 를 통한 해킹을 막을수있다.

214 ( CSRF 설명:

http://tm-csc.tistory.com/entry/7-WebHacking%EC%9B%B9%ED%95%B4%ED%82%B9-CSRF)

#### 217 2)/views/security/loginForm.jsp

```

218
219 <%@ page language="java" contentType="text/html; charset=UTF-8"

```

```

220     pageEncoding="UTF-8"%>
221 <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
222 <!DOCTYPE html>
223 <html>
224 <head>
225 <meta charset="UTF-8">
226 <title>로그인 창</title>
227 </head>
228 <body>
229     <h1>loginForm.jsp</h1>
230
231     <form action="<c:url value="j_spring_security_check" />" method="post">
232         ID : <input type="text" name="j_username"> <br />
233         PW : <input type="text" name="j_password"> <br />
234         <input type="submit" value="LOGIN"> <br />
235     </form>
236 </body>
237 </html>

```

### 3) HomeController.java 코드 추가

```

240 @RequestMapping("/loginForm.html")
241 public String loginForm(Model model) {
242     return "security/loginForm";
243 }

```

### 4) Test

```

246 -project > right-click > Run As > Run on Server
247 -http://localhost:8080/biz/login.html 을 요청하면
    http://localhost:8080/biz/loginForm.html 로 redirect
248 -아이디와 패스워드를 미리 설정한 javaexpert/12345678을 넣으면 정상적으로 login.jsp로 이동한다.
249 -http://localhost:8080/biz/welcome.html 을 요청하면
    http://localhost:8080/biz/loginForm.html 로 redirect
250 -만일 아이디와 패스워드를 미리 설정한 javaexpert/12345678을 넣으면 403 forbidden 에러가
    발생한다.
251 -만일 아이디와 패스워드를 미리 설정한 admin/12345678을 넣으면 정상적으로 welcome.jsp로
    이동한다.
252 -만일 잘못된 아이디와 패스워드를 입력하면 계속 loginForm.html?error가 나타난다.
253
254 -만일 Could not verify the provided CSRF token because your session was not found.
    오류가 발생하면 아래의 조치를 취한다.

```

```

255 --securityContext.xml 에서 코드 추가한다.
256 <security:http auto-config="true">
257     <security:form-login login-page="/loginForm.html" />
258     <security:intercept-url pattern="/login.html"
259         access="hasRole('ROLE_USER')" />
260     <security:intercept-url
261         pattern="/welcome.html" access="hasRole('ROLE_ADMIN')" />
262     <security:csrf disabled="true"/>        <---코드 추가
263 </security:http>

```

### 4. login 실패시 실패 메시지 출력하기

#### 1) securityContext.xml 코드 추가

```

269 <security:form-login login-page="/loginForm.html"
270     username-parameter="j_username"
271     password-parameter="j_password"
272     authentication-failure-url="/loginForm.html?ng" <---코드 추가
273     login-processing-url="/j_spring_security_check" />
274

```

## 2)loginForm.jsp 코드 추가

```
<h1>loginForm.jsp</h1>
<c:url value="j_spring_security_check" var="loginUrl" /> <!--추가
<form action="${loginUrl}" method="post"> <!-- 수정
  <c:if test="${param.ng != null}">
    <p>
      Login Failure<br />
      <c:if test="${SPRING_SECURITY_LAST_EXCEPTION != null}">
        Message : <c:out
          value="${SPRING_SECURITY_LAST_EXCEPTION.message}" />
      </c:if>
    </p>
  </c:if>
  ID : <input type="text" name="j_username"> <br />
  PW : <input type="text" name="j_password"> <br />
  <input type="submit" value="LOGIN"> <br />
</form>
```

## 3)Test

-http://localhost:8080/biz/login.html 을 요청하면  
http://localhost:8080/biz/loginForm.html 로 redirect  
-아이디와 패스워드를 미리 설정한 javaexpert/12345678을 넣으면 정상적으로 login.jsp로 이동한다.  
-만일 잘못된 아이디나 패스워드를 입력하면 아래와 같은 오류 메시지가 나온다.

```
Login Failure
Message : Bad credentials
```

-http://localhost:8080/biz/welcome.html 을 요청하면  
http://localhost:8080/biz/loginForm.html 로 redirect  
-만일 아이디와 패스워드를 미리 설정한 javaexpert/12345678을 넣으면 403 forbidden 에러가 발생한다.  
-만일 아이디와 패스워드를 미리 설정한 admin/12345678을 넣으면 정상적으로 welcome.jsp로 이동한다.  
-만일 잘못된 아이디와 패스워드를 입력하면 계속 loginForm.html?ng가 나오면서 아래의 메시지가 나타난다.

```
Login Failure
Message : Bad credentials
```

## 5. login, logout 상태 표시하기

### 1)login.jsp 페이지 코드 추가

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<body>
  <h1>login.jsp</h1>

  <c:if test="${not empty pageContext.request.userPrincipal}">
    <p>현재 로그인 상태</p>
  </c:if>
  <c:if test="${empty pageContext.request.userPrincipal}">
    <p>현재 로그아웃 상태</p>
  </c:if>

  User ID : ${pageContext.request.userPrincipal}<br />
  <a href="${pageContext.request.contextPath}/">Log Out</a>
</body>
```

## 2)Test

```

329 -http://localhost:8080/biz/login.html 을 요청하면
    http://localhost:8080/biz/loginForm.html 로 redirect
330 -아이디와 패스워드를 미리 설정한 javaexpert/12345678을 넣으면 정상적으로 login.jsp로 이동한다.
331 -이 때 화면에는 아래의 메시지가 나온다.
332
333     현재 로그인 상태
334     User ID :
        org.springframework.security.authentication.UsernamePasswordAuthenticationToken
        @b09a9234:
335     Principal: org.springframework.security.core.userdetails.User@f486f6cc:
336     Username: javaexpert;
337     Password: [PROTECTED];
338     Enabled: true;
339     AccountNonExpired: true;
340     credentialsNonExpired: true;
341     AccountNonLocked: true;
342     Granted Authorities: ROLE_USER;
343     Credentials: [PROTECTED];
344     Authenticated: true;
345     Details:
        org.springframework.security.web.authentication.WebAuthenticationDetails@166c8:
346     RemoteIpAddress: 0:0:0:0:0:0:0:1;
347     SessionId: 949A2731FFC91E3C350818E0772C8E45;
348     Granted Authorities: ROLE_USER
349
350     Log Out
351
352 3)여기서 username만 출력하려면
353
354     User ID : ${pageContext.request.userPrincipal.name} 으로 변경한다.
355
356 4)securityContext.xml 코드 추가
357
358     <security:form-login login-page="/loginForm.html"
359         username-parameter="j_username"
360         password-parameter="j_password"
361         authentication-failure-url="/loginForm.html?ng"
362         login-processing-url="/j_spring_security_check" />
363     <security:logout logout-url="/j_spring_security_logout"/>    <--코드 추가
364
365 5)Test
366     -로그인 창에서 Log Out link를 클릭하면 index.html로 이동한다.
367
368
369 6. Lab
370 1)Spring Legacy Project 생성
371     -In Package Explorer > right-click > New > Spring Legacy Project
372     -Project name : SpringSecurityDemo
373     -Select Spring MVC Project > Next
374     -com.javasoft.biz > Finish
375
376 2)server.xml에 project 추가
377     -Tomcat v8.5 Server at localhost > right-click > Add and Remove
378     -SpringSecurityDemo > Add > Finish > OK
379
380 3)Spring security library download and install
381     -In pom.xml 아래와 같이 수정
382         <java-version>1.8</java-version>
383         <org.springframework-version>4.3.17.RELEASE</org.springframework-version>
384

```

```
385 -In mvnrepository, search 'spring security'
386 -Select 'Spring Security Core' 4.2.6
387 -Select 'Spring Security Web' 4.2.6
388 -Select 'Spring Security Config' 4.2.6
389 -Select 'Spring Security Taglibs' 4.2.6
390 -pom.xml > right-click > Run As > Maven clean and Maven install
391
```

#### 392 4)HomeController.java 코드 수정

```
393
394 package com.javasoft.biz;
395
396 import java.util.Date;
397
398 import org.springframework.stereotype.Controller;
399 import org.springframework.ui.Model;
400 import org.springframework.web.bind.annotation.RequestMapping;
401 import org.springframework.web.bind.annotation.RequestMethod;
402
403 @Controller
404 public class HomeController {
405
406     @RequestMapping(value = "/", method = RequestMethod.GET)
407     public String home(Model model) {
408         model.addAttribute("serverTime", new Date());
409         return "home";
410     }
411 }
412
```

#### 413 5)project > right-click > Run As > Run on Server

#### 414 6)src/main/webapp/WEB-INF/spring/spring-security.xml 파일 생성

```
415 -check 'security'
416
```

#### 417 7)web.xml 코드 수정

```
418
419
420 <context-param>
421     <param-name>contextConfigLocation</param-name>
422     <param-value>/WEB-INF/spring/spring-security.xml</param-value>
423 </context-param>
424 ...
425 ...
426 <filter>
427     <filter-name>springSecurityFilterChain</filter-name>
428     <filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>
429 </filter>
430
431 <filter-mapping>
432     <filter-name>springSecurityFilterChain</filter-name>
433     <url-pattern>/*</url-pattern>
434 </filter-mapping>
435
```

#### 436 8)spring-security.xml

```
437
438 <?xml version="1.0" encoding="UTF-8"?>
439 <beans xmlns="http://www.springframework.org/schema/beans"
440     xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
441     xmlns:security="http://www.springframework.org/schema/security"
442     xsi:schemaLocation="http://www.springframework.org/schema/security
443         http://www.springframework.org/schema/security/spring-security-4.2.xsd
444         http://www.springframework.org/schema/beans
```



<http://www.springframework.org/schema/beans/spring-beans.xsd>">

```
444
445 <security:http auto-config="true" use-expressions="true">
446   <security:intercept-url pattern="/login"
447     access="permitAll" />
448   <security:intercept-url pattern="resources/**"
449     access="permitAll" />
450   <security:intercept-url pattern="/**"
451     access="hasRole('ROLE_USER')" />
452   <security:form-login login-page="/login"
453     default-target-url="/welcome" username-parameter="j_username"
454     password-parameter="j_password"
455     authentication-failure-url="/login?error"
456     login-processing-url="/j_spring_security_check"
457     always-use-default-target="true" />
458   <security:logout invalidate-session="true"
459     logout-url="/logout" logout-success-url="/login?logout" />
460   <security:csrf />
461 </security:http>
462
463 <security:authentication-manager>
464   <!-- <security:authentication-provider>
465     <security:user-service>
466       <security:user name="javaexpert" password="12345678"
467         authorities="ROLE_USER" />
468       <security:user name="admin" password="12345678"
469         authorities="ROLE_ADMIN,ROLE_USER" />
470     </security:user-service>
471   </security:authentication-provider> -->
472   <security:authentication-provider user-service-ref="memberService" />
473 </security:authentication-manager>
474
475 <bean id="memberService"
476   class="com.javasoft.service.MemberService">
477   <property name="userService" ref="userService" />
478 </bean>
479 <bean id="userService"
480   class="com.javasoft.service.UserService">
481   <property name="userDao" ref="userDao" />
482 </bean>
483 <bean id="userDao" class="com.javasoft.dao.UserDao">
484   <property name="username" value="javaexpert" />
485   <property name="password" value="12345678" />
486 </bean>
487 </beans>
```

## 9)HomeController.java

```
490 package com.javasoft.biz;
491
492 import java.util.Date;
493
494 import org.springframework.stereotype.Controller;
495 import org.springframework.ui.Model;
496 import org.springframework.web.bind.annotation.RequestMapping;
497 import org.springframework.web.bind.annotation.RequestMethod;
498 import org.springframework.web.bind.annotation.RequestParam;
499 import org.springframework.web.servlet.ModelAndView;
500
501 @Controller
502
```

```

503 public class HomeController {
504
505     @RequestMapping(value = "/", method = RequestMethod.GET)
506     public String home(Model model) {
507         model.addAttribute("serverTime", new Date());
508         return "home";
509     }
510
511     @RequestMapping(value = "/login", method = RequestMethod.GET)
512     public ModelAndView login(
513         @RequestParam(value = "error", required = false) String error,
514         @RequestParam(value = "logout", required = false) String logout) {
515         ModelAndView model = new ModelAndView();
516         if (error != null) {
517             model.addObject("error", "Invalid username and password!");
518         }
519         if (logout != null) {
520             model.addObject("msg", "You've been logged out successfully.");
521         }
522         model.setViewName("login");
523         return model;
524     }
525
526     @RequestMapping("/welcome")
527     public String welcome(Model model) {
528         return "welcome";
529     }
530
531     @RequestMapping(value = "/logout", method = RequestMethod.GET)
532     public String logout(Model model) {
533         return "logout";
534     }
535 }

```

10)com.javasoft.vo.UserVO.java

```

539 package com.javasoft.vo;
540
541 public class UserVO {
542     private String username;
543     private String password;
544
545     public UserVO() {}
546
547     public UserVO(String username, String password) {
548         this.username = username;
549         this.password = password;
550     }
551
552     public String getUsername() {
553         return username;
554     }
555
556     public void setUsername(String username) {
557         this.username = username;
558     }
559
560     public String getPassword() {
561         return password;
562     }

```

```

563     }
564
565     public void setPassword(String password) {
566         this.password = password;
567     }
568
569     @Override
570     public String toString() {
571         return "UserVO [username=" + username + ", password=" + password + "];"
572     }
573 }
574

```

11)com.javasoft.dao.UserDao.java

```

576
577     package com.javasoft.dao;
578
579     import org.springframework.beans.factory.annotation.Autowired;
580     import org.springframework.stereotype.Repository;
581
582     import com.javasoft.vo.UserVO;
583
584     @Repository("userDao")
585     public class UserDao {
586         @Autowired
587         private UserVO userVO;
588         private String username;
589         private String password;
590
591         public void setUsername(String username) {
592             this.username = username;
593         }
594
595         public void setPassword(String password) {
596             this.password = password;
597         }
598
599         public UserVO getUsersByID(String username) {
600             return new UserVO(this.username, this.password);
601         }
602     }
603

```

12)com.javasoft.service.UserService.java

```

605
606     package com.javasoft.service;
607
608     import org.springframework.beans.factory.annotation.Autowired;
609     import org.springframework.stereotype.Service;
610
611     import com.javasoft.dao.UserDao;
612     import com.javasoft.vo.UserVO;
613
614     public class UserService {
615         private UserDao userDao;
616
617         public void setUserDao(UserDao userDao) {
618             this.userDao = userDao;
619         }
620
621         public UserVO getUsersByID(String username) {
622             return this.userDao.getUsersByID(username);

```

```

623     }
624 }
625
626 13)com.javasoft.service.MemberService.java
627
628     package com.javasoft.service;
629
630     import java.util.ArrayList;
631     import java.util.Collection;
632
633     import org.springframework.beans.factory.annotation.Autowired;
634     import org.springframework.security.core.authority.SimpleGrantedAuthority;
635     import org.springframework.security.core.userdetails.User;
636     import org.springframework.security.core.userdetails.UserDetails;
637     import org.springframework.security.core.userdetails.UserDetailsService;
638     import org.springframework.security.core.userdetails.UsernameNotFoundException;
639     import org.springframework.stereotype.Service;
640
641     import com.javasoft.vo.UserVO;
642
643     public class MemberService implements UserDetailsService{
644         private UserService userService;
645
646
647         /*
648         * 사용자가 입력한 ID/PWD 를 검증하기위해 저장소에 그 ID/PWD 가 있는지 확인하는 소스이다.
649         * 저장소는 인메모리가 될 수도있고,
650         * DB가 될 수도있고 어떠한 것(페이스북,네이버등등)도 될 수있다. userService는 DB 에
651         * 접근하기위한 서비스이다.
652         */
653
654         public void setUserService(UserService userService) {
655             this.userService = userService;
656         }
657
658         @Override
659         public UserDetails loadUserByUsername(String username) throws
660             UsernameNotFoundException {
661             UserVO userVO = this.userService getUsersByID(username);
662             if(userVO == null)
663                 throw new UsernameNotFoundException("No user found with username " +
664                     userVO.getUsername());
665             Collection<SimpleGrantedAuthority> roles = new
666                 ArrayList<SimpleGrantedAuthority>();
667             roles.add(new SimpleGrantedAuthority("ROLE_USER"));
668             UserDetails user = new User(username, userVO.getPassword(), roles);
669             return user;
670         }
671     }
672
673 14)views/login.jsp
674
675     <%@ page language="java" contentType="text/html; charset=UTF-8"
676     pageEncoding="UTF-8"%>
677     <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
678     <!DOCTYPE html>
679     <html>
680     <head>
681         <title>로그인 페이지</title>
682         <meta charset="UTF-8">

```

```

678     <script>
679         function doLogin() {
680             if(frm.j_username.value == "") {
681                 alert("아이디를 입력해주세요.");
682                 return;
683             }
684             if(frm.j_password.value == "") {
685                 alert("패스워드를 입력해주세요.");
686                 return;
687             }
688             frm.submit();
689         }
690     </script>
691 </head>
692 <body>
693     <form name="frm" action="j_spring_security_check" method="post">
694         <ul>
695             <li>
696                 <label for="userID">ID</label>
697                 <input id = "userID" type="text" name="j_username" placeholder="ID"
698                     required>
699             </li>
700             <li>
701                 <label for="password">Password</label>
702                 <input id = "password" type="password" name="j_password"
703                     placeholder="PASSWORD" required>
704             </li>
705             <li>
706                 <input type="button" value="로그인" onclick="doLogin()"/>
707             </li>
708         </ul>
709         <input type="hidden" name="${_csrf.parameterName}"
710             value="${_csrf.token}" />
711     </form>
712
713     <c:if test="${not empty error}">
714         <div class="error">${error}</div>
715     </c:if>
716     <c:if test="${not empty msg}">
717         <div class="msg">${msg}</div>
718     </c:if>
719 </body>
720 </html>

```

#### 15) views/welcome.jsp

```

721 <%@ page language="java" contentType="text/html; charset=UTF-8"
722     pageEncoding="UTF-8"%>
723 <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
724 <!DOCTYPE html>
725 <html>
726 <head>
727     <meta charset="UTF-8">
728     <title>Member Page</title>
729 </head>
730 <body>
731     <c:if test="${pageContext.request.userPrincipal.name != null}">
732         <h2>
733             Welcome : ${pageContext.request.userPrincipal.name} |
734             <a href="logout"> Logout</a>
735         </h2>

```

```

736     </c:if>
737 </body>
738 </html>
739
740 16)logout.jsp
741
742 <%@ page language="java" contentType="text/html; charset=UTF-8"
743     pageEncoding="UTF-8"%>
744 <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
745 <c:set var="username" value="${pageContext.request.userPrincipal.name}" />
746 <!DOCTYPE html>
747 <html>
748 <head>
749 <meta charset="UTF-8">
750 <title>Logout page</title>
751 </head>
752 <body>
753     <c:url value="/logout" var="logoutUrl" />
754     <!-- csrt for log out-->
755     <form action="${logoutUrl}" method="post" id="logoutForm">
756         <input type="hidden" name="${_csrf.parameterName}"
757             value="${_csrf.token}" />
758     </form>
759     <script>
760         function formSubmit() {
761             document.getElementById("logoutForm").submit();
762         }
763     </script>
764     <form>
765         <span style="color: gray;" ><h5> ${username} 님 반갑습니다.
766         <a href = "javascript:formSubmit()"> 로그아웃 </a> </h5></span>
767     </form>
768 </body>
769 </html>
770
771

```

## 772 7. 보안관련 taglibs 사용하기

773 1)pom.xml 에서 spring-security-taglibs : 4.2.6.RELEASE 확인

774 2)login.jsp에서

775 -아래 taglib 추가

```

776 <%@ taglib uri="http://www.springframework.org/security/tags" prefix="s" %>
777

```

778 -코드 변경

```

780 <c:if test="${not empty pageContext.request.userPrincipal}">
781     <p>현재 로그인 상태</p>
782 </c:if>
783 <c:if test="${empty pageContext.request.userPrincipal}">
784     <p>현재 로그아웃 상태</p>
785 </c:if>
786

```

787 -위의 코드를 아래의 코드로 변경

```

788
789 <s:authorize access="hasRole('ROLE_USER')">
790     <p>현재 로그인 상태</p>
791 </s:authorize>
792 <s:authorize access="hasRole('ROLE_ANONYMOUS')">
793     <p>현재 로그아웃 상태</p>
794 </s:authorize>
795

```

-코드 변경

```
User ID : ${pageContext.request.userPrincipal.name}<br />
```

-아래의 코드로 변경

```
User ID : <s:authentication property="name" /><br />
```

## 8. Database-based Login Authentication

### 1)Spring Legacy Project 생성

- In Package Explorer > right-click > New > Spring Legacy Project
- Project name : SpringSecurityDatabaseDemo
- Select Spring MVC Project > Next
- com.javasoft.biz > Finish

### 2)server.xml에 project 추가

- Tomcat v8.5 Server at localhost > right-click > Add and Remove
- SpringSecurityDatabaseDemo > Add > Finish > OK

### 3)Spring security library download and install

- In pom.xml 아래와 같이 수정

```
<java-version>1.8</java-version>
<org.springframework-version>4.3.17.RELEASE</org.springframework-version>
```
- In mvnrepository, search 'spring security'
- Select 'Spring Security Core' 4.2.6
- Select 'Spring Security Web' 4.2.6
- Select 'Spring Security Config' 4.2.6
- Select 'Spring Security Taglibs' 4.2.6
- pom.xml > right-click > Run As > Maven clean and Maven install

### 4)HomeController.java

```
package com.javasoft.biz;

import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;

@Controller
public class HomeController {
    @RequestMapping(value = "/", method = RequestMethod.GET)
    public String home(Model model) {
        model.addAttribute("greeting", "Hello Spring Security");
        return "home";
    }
}
```

### 5)home.jsp

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ page session="false" %>
<html>
<head>
    <title>Home</title>
</head>
<body>
<h1>${greeting} </h1>
```

```
856
857     </body>
858 </html>
859
```

860 6)project > right-click > Run As > Run on Server

861 7)Database table creation

```
862
863
864     CREATE SEQUENCE seq_user_role_id
865     MAXVALUE 9999;
866     /
867
868     CREATE TABLE Users (
869         username VARCHAR2(20),
870         password VARCHAR2(20) NOT NULL,
871         enabled NUMBER(1) DEFAULT 1 ,
872         CONSTRAINT users_username_pk PRIMARY KEY (username)
873     );
874
875     CREATE TABLE User_roles (
876         user_role_id NUMBER(4),
877         username VARCHAR2(45),
878         role VARCHAR2(45) NOT NULL,
879         CONSTRAINT user_roles_id PRIMARY KEY (user_role_id),
880         CONSTRAINT user_roles_uk UNIQUE (role,username),
881         CONSTRAINT user_roles_fk FOREIGN KEY (username) REFERENCES users
882         (username)
883     );
884
```

885 8)Inserts some records for testing.

```
886     INSERT INTO Users(username,password,enabled) VALUES ('javaexpert','12345678', 1);
887     INSERT INTO users(username,password,enabled) VALUES ('alex','12345678', 1);
888
889     INSERT INTO user_roles (user_role_id, username, role)
890     VALUES (seq_user_role_id.NEXTVAL, 'javaexpert', 'ROLE_USER');
891
892     INSERT INTO user_roles (user_role_id, username, role)
893     VALUES (seq_user_role_id.NEXTVAL, 'javaexpert', 'ROLE_ADMIN');
894
```

895 9)pom.xml에 oracle driver와 MyBatis 관련 dependency 추가

```
896
897     <dependency>
898         <groupId>com.oracle</groupId>
899         <artifactId>ojdbc8</artifactId>
900         <version>12.2</version>
901     </dependency>
902     <!-- https://mvnrepository.com/artifact/org.mybatis/mybatis -->
903     <dependency>
904         <groupId>org.mybatis</groupId>
905         <artifactId>mybatis</artifactId>
906         <version>3.4.6</version>
907     </dependency>
908     <!-- https://mvnrepository.com/artifact/org.mybatis/mybatis-spring -->
909     <dependency>
910         <groupId>org.mybatis</groupId>
911         <artifactId>mybatis-spring</artifactId>
912         <version>1.3.2</version>
913     </dependency>
914
```



## 10) Database 연결을 위한 환경설정

-pom.xml에 spring-jdbc 추가

```
<!-- https://mvnrepository.com/artifact/org.springframework/spring-jdbc -->
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-jdbc</artifactId>
  <version>4.3.17.RELEASE</version>
</dependency>
```

-web.xml 코드 추가

```
<filter>
  <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>
```

-web.xml 수정

```
<context-param>
  <param-name>contextConfigLocation</param-name>
  <param-value>/WEB-INF/spring/spring-security.xml</param-value>
</context-param>
```

-src/main/webapp/WEB-INF/spring/spring-security.xml 생성

```
--src/main/webapp/WEB-INF/spring > right-click > New > Spring Bean
Configuration File
--File name : spring-security.xml > Next
--Check beans, security > Finish
```

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns:security="http://www.springframework.org/schema/security"
  xsi:schemaLocation="http://www.springframework.org/schema/security
    http://www.springframework.org/schema/security/spring-security-4.2.xsd
    http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans.xsd">

  <bean id="dataSource"
    class="org.springframework.jdbc.datasource.DriverManagerDataSource">
    <property name="driverClassName"
      value="oracle.jdbc.driver.OracleDriver" />
    <property name="url"
      value="jdbc:oracle:thin:@192.168.56.3:1521:ORCL" />
    <property name="username" value="scott" />
    <property name="password" value="tiger" />
  </bean>

  <!-- enable use-expressions -->
  <security:http>

    <security:intercept-url pattern="/admin**"
      access="hasRole('ROLE_ADMIN')" />
```

```

969         <!-- access denied page -->
970         <security:access-denied-handler error-page="/403" />
971
972         <security:form-login
973             login-page="/login"
974             default-target-url="/welcome"
975             authentication-failure-url="/login?error"
976             login-processing-url="/j_spring_security_check"
977             username-parameter="username"
978             password-parameter="password" />
979         <security:logout invalidate-session="true"
980             logout-url="/j_spring_security_logout"
981             logout-success-url="/login?logout" />
982         <!-- enable csrf protection -->
983         <security:csrf disabled="true"/>
984     </security:http>
985
986     <!-- Select users and user_roles from database -->
987     <security:authentication-manager>
988         <security:authentication-provider>
989             <security:jdbc-user-service data-source-ref="dataSource"
990                 users-by-username-query=
991                     "select username,password, enabled from users where username=?"
992                 authorities-by-username-query=
993                     "select username, role from user_roles where username =? " />
994         </security:authentication-provider>
995     </security:authentication-manager>
996 </beans>

```

## 11)views/hello.jsp

```

999     <%@ page language="java" contentType="text/html; charset=UTF-8"
1000         pageEncoding="UTF-8"%>
1001     <%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
1002     <%@taglib prefix="sec"
1003         uri="http://www.springframework.org/security/tags"%>
1004     <html>
1005     <body>
1006         <h1>Title : ${title}</h1>
1007         <h1>Message : ${message}</h1>
1008
1009         <sec:authorize access="hasRole('ROLE_USER')">
1010             <!-- For login user -->
1011             <c:url value="/j_spring_security_logout" var="logoutUrl" />
1012             <form action="${logoutUrl}" method="post" id="logoutForm">
1013                 <input type="hidden" name="${_csrf.parameterName}"
1014                     value="${_csrf.token}" />
1015             </form>
1016             <script>
1017                 function formSubmit() {
1018                     document.getElementById("logoutForm").submit();
1019                 }
1020             </script>
1021
1022             <c:if test="${pageContext.request.userPrincipal.name != null}">
1023                 <h2>
1024                     User : ${pageContext.request.userPrincipal.name} | <a
1025                         href="javascript:formSubmit()"> Logout</a>
1026                 </h2>
1027             </c:if>

```

```

1028
1029
1030     </sec:authorize>
1031 </body>
1032 </html>
1033
1034 12)views/login.jsp
1035
1036 <%@ page language="java" contentType="text/html; charset=UTF-8"
1037     pageEncoding="UTF-8"%>
1038     <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
1039 <html>
1040 <head>
1041 <title>Login Page</title>
1042 <style>
1043     .error {
1044         padding: 15px;
1045         margin-bottom: 20px;
1046         border: 1px solid transparent;
1047         border-radius: 4px;
1048         color: #a94442;
1049         background-color: #f2dede;
1050         border-color: #ebccd1;
1051     }
1052
1053     .msg {
1054         padding: 15px;
1055         margin-bottom: 20px;
1056         border: 1px solid transparent;
1057         border-radius: 4px;
1058         color: #31708f;
1059         background-color: #d9edf7;
1060         border-color: #bce8f1;
1061     }
1062
1063     #login-box {
1064         width: 300px;
1065         padding: 20px;
1066         margin: 100px auto;
1067         background: #fff;
1068         -webkit-border-radius: 2px;
1069         -moz-border-radius: 2px;
1070         border: 1px solid #000;
1071     }
1072 </style>
1073 </head>
1074 <body onload='document.loginForm.username.focus();'>
1075
1076     <h1>Spring Security Login Form (Database Authentication)</h1>
1077
1078     <div id="login-box">
1079
1080         <h2>Login with Username and Password</h2>
1081
1082         <c:if test="{not empty error}">
1083             <div class="error">${error}</div>
1084         </c:if>
1085         <c:if test="{not empty msg}">
1086             <div class="msg">${msg}</div>
1087         </c:if>

```

```

1088
1089     <form name='loginForm'
1090         action="<c:url value='/j_spring_security_check' />" method='POST'>
1091
1092     <table>
1093         <tr>
1094             <td>User: </td>
1095             <td><input type='text' name='username'> </td>
1096         </tr>
1097         <tr>
1098             <td>Password: </td>
1099             <td><input type='password' name='password' /> </td>
1100         </tr>
1101         <tr>
1102             <td colspan='2'><input name="submit" type="submit"
1103                 value="submit" /> </td>
1104         </tr>
1105     </table>
1106
1107     <input type="hidden" name="_${_csrf.parameterName}"
1108         value="_${_csrf.token}" />
1109
1110 </form>
1111 </div>
1112 </body>
1113 </html>

```

### 13)views/admin.jsp

```

1116
1117 <%@ page language="java" contentType="text/html; charset=UTF-8"
1118     pageEncoding="UTF-8"%>
1119 <%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
1120 <html>
1121 <body>
1122     <h1>Title : ${title}</h1>
1123     <h1>Message : ${message}</h1>
1124
1125     <c:url value="/j_spring_security_logout" var="logoutUrl" />
1126     <form action="${logoutUrl}" method="post" id="logoutForm">
1127         <input type="hidden" name="_${_csrf.parameterName}"
1128             value="_${_csrf.token}" />
1129     </form>
1130     <script>
1131         function formSubmit() {
1132             document.getElementById("logoutForm").submit();
1133         }
1134     </script>
1135
1136     <c:if test="${pageContext.request.userPrincipal.name != null}">
1137         <h2>
1138             Welcome : ${pageContext.request.userPrincipal.name} | <a
1139                 href="javascript:formSubmit()"> Logout</a>
1140         </h2>
1141     </c:if>
1142
1143 </body>
1144 </html>

```

### 14)views/403.jsp

```

1148 <%@ page language="java" contentType="text/html; charset=UTF-8"
1149     pageEncoding="UTF-8"%>
1150 <%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
1151 <html>
1152 <body>
1153     <h1>HTTP Status 403 - Access is denied</h1>
1154
1155     <c:choose>
1156         <c:when test="${empty username}">
1157             <h2>You do not have permission to access this page!</h2>
1158         </c:when>
1159         <c:otherwise>
1160             <h2>Username : ${username} <br/>
1161                 You do not have permission to access this page!</h2>
1162         </c:otherwise>
1163     </c:choose>
1164
1165 </body>
1166 </html>
1167

```

#### 15)HomeController.java

```

1169
1170 @RequestMapping(value = { "/", "/welcome**" }, method = RequestMethod.GET)
1171 public ModelAndView defaultPage() {
1172
1173     ModelAndView model = new ModelAndView();
1174     model.addObject("title", "Spring Security Login Form - Database Authentication");
1175     model.addObject("message", "This is default page!");
1176     model.setViewName("hello");
1177     return model;
1178
1179 }
1180
1181 @RequestMapping(value = "/admin**", method = RequestMethod.GET)
1182 public ModelAndView adminPage() {
1183
1184     ModelAndView model = new ModelAndView();
1185     model.addObject("title", "Spring Security Login Form - Database Authentication");
1186     model.addObject("message", "This page is for ROLE_ADMIN only!");
1187     model.setViewName("admin");
1188     return model;
1189
1190 }
1191
1192 @RequestMapping(value = "/login", method = RequestMethod.GET)
1193 public ModelAndView login(@RequestParam(value = "error", required = false) String
error,
1194     @RequestParam(value = "logout", required = false) String logout) {
1195
1196     ModelAndView model = new ModelAndView();
1197     if (error != null) {
1198         model.addObject("error", "Invalid username and password!");
1199     }
1200
1201     if (logout != null) {
1202         model.addObject("msg", "You've been logged out successfully.");
1203     }
1204     model.setViewName("login");
1205
1206     return model;

```

```
1207
1208     }
1209
1210     //for 403 access denied page
1211     @RequestMapping(value = "/403", method = RequestMethod.GET)
1212     public ModelAndView accesssDenied() {
1213
1214         ModelAndView model = new ModelAndView();
1215
1216         //check if user is login
1217         Authentication auth = SecurityContextHolder.getContext().getAuthentication();
1218         if (!(auth instanceof AnonymousAuthenticationToken)) {
1219             UserDetails userDetail = (UserDetails) auth.getPrincipal();
1220             model.addObject("username", userDetail.getUsername());
1221         }
1222
1223         model.setViewName("403");
1224         return model;
1225
1226     }
1227     @RequestMapping(value = "/logout", method = RequestMethod.GET)
1228     public String logout(Model model) {
1229         return "logout";
1230     }
1231
```