

```
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
21  11)개별 사용자에게 대해 일일이 권한을 가졌는지를 검사하는 것을 비효율적일 뿐만 아니라 경우에 따라서는 불가능하기도
22  12)따라서, 사용자에게 권한을 부여할 때 가장 많이 사용하는 일반적인 방법은 사용자를 역할(role)에 할당하는 역할
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
61 <context-param>
62     <param-name>contextConfigLocation</param-name>
63     <param-value>classpath:securityContext.xml</param-value>
64 </context-param>
65
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.
90
91 <security:authentication-manager>
92     <security:authentication-provider>
93         <security:user-service>
94             <security:user name="javaexpert" password="12345678" authorities="ROLE_USER"/>
95             <security:user name="admin" password="12345678" authorities="ROLE_ADMIN,ROLE_USER"/>
96         </security:user-service>
97     </security:authentication-provider>
98 </security:authentication-manager>
99
100 ---<authentication-manager> ---인증처리를 위한 최상위 태그
101     <authentication-provider user-service-ref="memberService"/> ----사용자이름/비밀번호를
102     </authentication-manager>

```

```
103      ---위의 예제는 Inmemory로 처리하고 있음.
104
105
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
119      -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
126      6) Test
127      -/WEB-INF/views/security folder 생성
128      -/WEB-INF/views/security/login.jsp
129
130      <body>
131          <h1>login.jsp</h1>
132      </body>
133
134      -/WEB-INF/views/security/welcome.jsp
135
136      <body>
137          <h1>welcome.jsp</h1>
138      </body>
139
140      -/WEB-INF/views/home.jsp
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
156 -HomeController.java 코드 수정
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.
184
185
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
203         default-target-url="/monitoring" -----로그인성공하면 이동할 페이지설정
204         username-parameter="username"

```

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

239 3)HomeController.java 코드 추가

245 4)Test

Page 5 of 25

```

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

```

268
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

```

276
277     <h1>loginForm.jsp</h1>
278     <c:url value="/j_spring_security_check" var="loginUrl" />  <--추가
279     <form action="${loginUrl}" method="post">                <-- 수정
280         <c:if test="${param.ng != null}">
281             <p>
282                 Login Failure<br />
283                 <c:if test="${SPRING_SECURITY_LAST_EXCEPTION != null}">
284                     Message : <c:out value="${SPRING_SECURITY_LAST_EXCEPTION.message}" />
285                 </c:if>
286             </p>
287         </c:if>
288         ID : <input type="text" name="j_username"> <br />
289         PW : <input type="text" name="j_password"> <br />
290         <input type="submit" value="LOGIN"> <br />
291     </form>
292

```

3)Test

```

294     -<a href="http://localhost:8080/biz/login.html">http://localhost:8080/biz/login.html</a> 을 요청하면 <a href="http://localhost:8080/biz/loginForm.html">http://localhost:8080/biz/loginForm.html</a>
295     -아이디와 패스워드를 미리 설정한 javaexpert/12345678을 넣으면 정상적으로 login.jsp로 이동한다.
296     -만일 잘못된 아이디나 패스워드를 입력하면 아래와 같은 오류 메시지가 나온다.
297
298     Login Failure
299     Message : Bad credentials
300
301     -<a href="http://localhost:8080/biz/welcome.html">http://localhost:8080/biz/welcome.html</a> 을 요청하면 <a href="http://localhost:8080/biz/loginForm.html">http://localhost:8080/biz/loginForm.html</a>
302     -만일 아이디와 패스워드를 미리 설정한 javaexpert/12345678을 넣으면 403 forbidden 에러가 발생한다.
303     -만일 아이디와 패스워드를 미리 설정한 admin/12345678을 넣으면 정상적으로 welcome.jsp로 이동한다.
304     -만일 잘못된 아이디와 패스워드를 입력하면 계속 loginForm.html?ng가 나오면서 아래의 메시지가 나타난다.
305
306     Login Failure

```

307 Message : Bad credentials

308

309

310 5. login, logout 상태 표시하기

311 1)login.jsp 페이지 코드 추가

312

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

314 <body>

315 <h1>login.jsp</h1>

316

317 <c:if test="\${not empty pageContext.request.userPrincipal}">

318 <p>현재 로그인 상태</p>

319 </c:if>

320 <c:if test="\${empty pageContext.request.userPrincipal}">

321 <p>현재 로그아웃 상태</p>

322 </c:if>

323

324 User ID : \${pageContext.request.userPrincipal}

325 Log Out

326 </body>

327

328 2)Test

329 -<http://localhost:8080/biz/login.html> 을 요청하면 <http://localhost:8080/biz/loginForm.html>

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](#)

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.

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>
```

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
415 6)src/main/webapp/WEB-INF/spring/spring-security.xml 파일 생성
416 -check 'security'

417
418 7)web.xml 코드 수정

```
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 http://www.springframework.org/schema/security
443     http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans">
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
491 package com.javasoft.biz;
492
493 import java.util.Date;
494
495 import org.springframework.stereotype.Controller;
496 import org.springframework.ui.Model;
497 import org.springframework.web.bind.annotation.RequestMapping;
498 import org.springframework.web.bind.annotation.RequestMethod;
499 import org.springframework.web.bind.annotation.RequestParam;
500 import org.springframework.web.servlet.ModelAndView;
501
502 @Controller
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 }
```

536
537
538 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() {
```

```
562         return password;
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
575 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
604 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
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663

```
public class UserService {
    private UserDao userDao;

    public void setUserDao(UserDao userDao) {
        this.userDao = userDao;
    }

    public UserVO getUsersByID(String username) {
        return this.userDao.getUsersByID(username);
    }
}

13)com.javasoft.service.MemberService.java

package com.javasoft.service;

import java.util.ArrayList;
import java.util.Collection;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.security.core.authority.SimpleGrantedAuthority;
import org.springframework.security.core.userdetails.User;
import org.springframework.security.core.userdetails.UserDetails;
import org.springframework.security.core.userdetails.UserDetailsService;
import org.springframework.security.core.userdetails.UsernameNotFoundException;
import org.springframework.stereotype.Service;

import com.javasoft.vo.UserVO;

public class MemberService implements UserDetailsService{
    private UserService userService;

    /*
     * 사용자가 입력한 ID/PWD 를 검증하기위해 저장소에 그 ID/PWD 가 있는지 확인하는 소스이다. 저장소는
     * DB가 될 수도있고 어떠한 것(페이스북,네이버등등)도 될 수있다. userService는 DB 에 접근하기위한 서비스이다
     */

    public void setUserService(UserService userService) {
        this.userService = userService;
    }

    @Override
    public UserDetails loadUserByUsername(String username) throws UsernameNotFoundException{
        UserVO userVO = this.userService.getUsersByID(username);
        if(userVO == null)
            throw new UsernameNotFoundException("No user found with username " + userVO.getUserName());
        Collection<SimpleGrantedAuthority> roles = new ArrayList<SimpleGrantedAuthority>();
        roles.add(new SimpleGrantedAuthority("ROLE_USER"));
        UserDetails user = new User(username, userVO.getPassword(), roles);
    }
}
```

```
664     return user;
665   }
666 }
667
668 14)views/login.jsp
669
670 <%@ page language="java" contentType="text/html; charset=UTF-8"
671     pageEncoding="UTF-8"%>
672 <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
673 <!DOCTYPE html>
674 <html>
675 <head>
676     <title>로그인 페이지</title>
677     <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" required>
698             </li>
699             <li>
700                 <label for="password">Password</label>
701                 <input id = "password" type="password" name="j_password" placeholder="PASSWORD">
702             </li>
703             <li>
704                 <input type="button" value="로그인" onclick="doLogin()"/>
705             </li>
706         </ul>
707         <input type="hidden" name="${_csrf.parameterName}"
708             value="${_csrf.token}" />
709     </form>
710
711     <c:if test="${not empty error}">
712         <div class="error">${error}</div>
713     </c:if>
714     <c:if test="${not empty msg}">
715         <div class="msg">${msg}</div>
716     </c:if>
```

```
715     </c:if>
716 </body>
717 </html>
718
719 15)views/welcome.jsp
720
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
779
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
796     -코드 변경
797
798     User ID : ${pageContext.request.userPrincipal.name}<br />
799
800     -아래의 코드로 변경
801
802     User ID : <s:authentication property="name" /><br />
803
804
805 8. Database-based Login Authentication
806 1)Spring Legacy Project 생성
807     -In Package Explorer > right-click > New > Spring Legacy Project
808     -Project name : SpringSecurityDatabaseDemo
809     -Select Spring MVC Project > Next
810     -com.javasoft.biz > Finish
811
812 2)server.xml에 project 추가
813     -Tomcat v8.5 Server at localhost > right-click > Add and Remove
814     -SpringSecurityDatabaseDemo > Add > Finish > OK
815
816 3)Spring security library download and install
```



```
817 -In pom.xml 아래와 같이 수정
818 <java-version>1.8</java-version>
819 <org.springframework-version>4.3.17.RELEASE</org.springframework-version>
820
821 -In mvnrepository, search 'spring security'
822 -Select 'Spring Security Core' 4.2.6
823 -Select 'Spring Security Web' 4.2.6
824 -Select 'Spring Security Config' 4.2.6
825 -Select 'Spring Security Taglibs' 4.2.6
826 -pom.xml > right-click > Run As > Maven clean and Maven install
827
```

828 4)HomeController.java

```
829
830 package com.javasoft.biz;
831
832 import org.springframework.stereotype.Controller;
833 import org.springframework.ui.Model;
834 import org.springframework.web.bind.annotation.RequestMapping;
835 import org.springframework.web.bind.annotation.RequestMethod;
836
837 @Controller
838 public class HomeController {
839     @RequestMapping(value = "/", method = RequestMethod.GET)
840     public String home(Model model) {
841         model.addAttribute("greeting", "Hello Spring Security");
842         return "home";
843     }
844 }
```

846 5)home.jsp

```
847
848 <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
849 <%@ page session="false" %>
850 <html>
851 <head>
852 <title>Home</title>
853 </head>
854 <body>
855 <h1>${greeting} </h1>
856
857 </body>
858 </html>
859
```

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

862 7)Database table creation

```
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 (username)  
882 );  
883
```

884 8)Inserts some records for testing.

```
885  
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
```

915 10)Database 연결을 위한 환경설정

916 -pom.xml에 spring-jdbc 추가

```
917  
918 <!-- https://mvnrepository.com/artifact/org.springframework/spring-jdbc -->
```

```
919     <dependency>
920         <groupId>org.springframework</groupId>
921         <artifactId>spring-jdbc</artifactId>
922         <version>4.3.17.RELEASE</version>
923     </dependency>
924
925 -web.xml 코드 추가
926
927     <filter>
928         <filter-name>springSecurityFilterChain</filter-name>
929         <filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>
930     </filter>
931
932     <filter-mapping>
933         <filter-name>springSecurityFilterChain</filter-name>
934         <url-pattern>/*</url-pattern>
935     </filter-mapping>
936
937 -web.xml 수정
938
939     <context-param>
940         <param-name>contextConfigLocation</param-name>
941         <param-value>/WEB-INF/spring/spring-security.xml</param-value>
942     </context-param>
943
944 -src/main/webapp/WEB-INF/spring/spring-security.xml 생성
945 --src/main/webapp/WEB-INF/spring > right-click > New > Spring Bean Configuration File
946 --File name : spring-security.xml > Next
947 --Check beans, security > Finish
948
949     <?xml version="1.0" encoding="UTF-8"?>
950     <beans xmlns="http://www.springframework.org/schema/beans"
951         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
952         xmlns:security="http://www.springframework.org/schema/security"
953         xsi:schemaLocation="http://www.springframework.org/schema/security http://www.springframework.org/schema/beans http://www.springframework.org/schema/security"
954     >
955
956         <bean id="dataSource"
957             class="org.springframework.jdbc.datasource.DriverManagerDataSource">
958             <property name="driverClassName" value="oracle.jdbc.driver.OracleDriver" />
959             <property name="url" value="jdbc:oracle:thin:@192.168.56.3:1521:ORCL" />
960             <property name="username" value="scott" />
961             <property name="password" value="tiger" />
962         </bean>
963
964         <!-- enable use-expressions -->
965         <security:http auto-config="true" use-expressions="true">
966
967             <security:intercept-url pattern="/admin*" access="hasRole('ROLE_ADMIN')" />
968
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             username-parameter="username"
977             password-parameter="password" />
978         <security:logout logout-success-url="/login?logout" />
979         <!-- enable csrf protection -->
980         <security:csrf />
981     </security:http>
982
983     <!-- Select users and user_roles from database -->
984     <security:authentication-manager>
985         <security:authentication-provider>
986             <security:jdbc-user-service data-source-ref="dataSource"
987                 users-by-username-query=
988                     "select username,password, enabled from users where username=?"
989                 authorities-by-username-query=
990                     "select username, role from user_roles where username =? " />
991             </security:authentication-provider>
992         </security:authentication-manager>
993     </beans>
994
995 11)views/hello.jsp
996
997     <%@ page language="java" contentType="text/html; charset=UTF-8"
998         pageEncoding="UTF-8"%>
999     <%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
1000     <%@taglib prefix="sec"
1001         uri="http://www.springframework.org/security/tags"%>
1002     <html>
1003     <body>
1004         <h1>Title : ${title}</h1>
1005         <h1>Message : ${message}</h1>
1006
1007         <sec:authorize access="hasRole('ROLE_USER')">
1008             <!-- For login user -->
1009             <c:url value="/j_spring_security_logout" var="logoutUrl" />
1010             <form action="${logoutUrl}" method="post" id="logoutForm">
1011                 <input type="hidden" name="${_csrf.parameterName}"
1012                     value="${_csrf.token}" />
1013             </form>
1014             <script>
1015                 function formSubmit() {
1016                     document.getElementById("logoutForm").submit();
1017                 }
1018             </script>
1019
1020             <c:if test="${pageContext.request.userPrincipal.name != null}">

```

```
1021         <h2>
1022             User : ${pageContext.request.userPrincipal.name} | <a
1023                 href="javascript:formSubmit()"> Logout</a>
1024         </h2>
1025     </c:if>
1026
1027
1028     </sec:authorize>
1029 </body>
1030 </html>
1031
1032 12)views/login.jsp
1033
1034 <%@ page language="java" contentType="text/html; charset=UTF-8"
1035     pageEncoding="UTF-8"%>
1036     <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
1037 <html>
1038 <head>
1039 <title>Login Page</title>
1040 <style>
1041     .error {
1042         padding: 15px;
1043         margin-bottom: 20px;
1044         border: 1px solid transparent;
1045         border-radius: 4px;
1046         color: #a94442;
1047         background-color: #f2dede;
1048         border-color: #ebccd1;
1049     }
1050
1051     .msg {
1052         padding: 15px;
1053         margin-bottom: 20px;
1054         border: 1px solid transparent;
1055         border-radius: 4px;
1056         color: #31708f;
1057         background-color: #d9edf7;
1058         border-color: #bce8f1;
1059     }
1060
1061     #login-box {
1062         width: 300px;
1063         padding: 20px;
1064         margin: 100px auto;
1065         background: #fff;
1066         -webkit-border-radius: 2px;
1067         -moz-border-radius: 2px;
1068         border: 1px solid #000;
1069     }
1070 </style>
1071 </head>
```

```
1072 <body onload='document.loginForm.username.focus();'>
1073
1074 <h1>Spring Security Login Form (Database Authentication)</h1>
1075
1076 <div id="login-box">
1077
1078 <h2>Login with Username and Password</h2>
1079
1080 <c:if test="${not empty error}">
1081 <div class="error">${error}</div>
1082 </c:if>
1083 <c:if test="${not empty msg}">
1084 <div class="msg">${msg}</div>
1085 </c:if>
1086
1087 <form name='loginForm'
1088 action="<c:url value='/j_spring_security_check' />" method='POST'>
1089
1090 <table>
1091 <tr>
1092 <td>User:</td>
1093 <td><input type='text' name='username'></td>
1094 </tr>
1095 <tr>
1096 <td>Password:</td>
1097 <td><input type='password' name='password' /></td>
1098 </tr>
1099 <tr>
1100 <td colspan='2'><input name="submit" type="submit"
1101 value="submit" /></td>
1102 </tr>
1103 </table>
1104
1105 <input type="hidden" name="${_csrf.parameterName}"
1106 value="${_csrf.token}" />
1107
1108 </form>
1109 </div>
1110 </body>
1111 </html>
```

1112 13)views/admin.jsp

```
1113
1114
1115 <%@ page language="java" contentType="text/html; charset=UTF-8"
1116 pageEncoding="UTF-8"%>
1117 <%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
1118 <html>
1119 <body>
1120 <h1>Title : ${title}</h1>
1121 <h1>Message : ${message}</h1>
1122
```

```
1123     <c:url value="/j_spring_security_logout" var="logoutUrl" />
1124     <form action="{logoutUrl}" method="post" id="logoutForm">
1125         <input type="hidden" name="{_csrf.parameterName}"
1126             value="{_csrf.token}" />
1127     </form>
1128     <script>
1129         function formSubmit() {
1130             document.getElementById("logoutForm").submit();
1131         }
1132     </script>
1133
1134     <c:if test="{pageContext.request.userPrincipal.name != null}">
1135         <h2>
1136             Welcome : {pageContext.request.userPrincipal.name} | <a
1137                 href="javascript:formSubmit()"> Logout</a>
1138         </h2>
1139     </c:if>
1140
1141 </body>
1142 </html>
```

14)views/403.jsp

```
1145
1146 <%@ page language="java" contentType="text/html; charset=UTF-8"
1147     pageEncoding="UTF-8"%>
1148 <%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
1149 <html>
1150 <body>
1151     <h1>HTTP Status 403 - Access is denied</h1>
1152
1153     <c:choose>
1154         <c:when test="{empty username}">
1155             <h2>You do not have permission to access this page!</h2>
1156         </c:when>
1157         <c:otherwise>
1158             <h2>Username : {username} <br/>
1159                 You do not have permission to access this page!</h2>
1160         </c:otherwise>
1161     </c:choose>
1162
1163 </body>
1164 </html>
```

15)HomeController.java

```
1165
1166
1167
1168 @RequestMapping(value = { "/", "/welcome*" }, method = RequestMethod.GET)
1169 public ModelAndView defaultPage() {
1170
1171     ModelAndView model = new ModelAndView();
1172     model.addObject("title", "Spring Security Login Form - Database Authentication");
1173     model.addObject("message", "This is default page!");
```

```
1174     model.setViewName("hello");
1175     return model;
1176
1177 }
1178
1179 @RequestMapping(value = "/admin**", method = RequestMethod.GET)
1180 public ModelAndView adminPage() {
1181
1182     ModelAndView model = new ModelAndView();
1183     model.addObject("title", "Spring Security Login Form - Database Authentication");
1184     model.addObject("message", "This page is for ROLE_ADMIN only!");
1185     model.setViewName("admin");
1186     return model;
1187
1188 }
1189
1190 @RequestMapping(value = "/login", method = RequestMethod.GET)
1191 public ModelAndView login(@RequestParam(value = "error", required = false) String error,
1192     @RequestParam(value = "logout", required = false) String logout) {
1193
1194     ModelAndView model = new ModelAndView();
1195     if (error != null) {
1196         model.addObject("error", "Invalid username and password!");
1197     }
1198
1199     if (logout != null) {
1200         model.addObject("msg", "You've been logged out successfully.");
1201     }
1202     model.setViewName("login");
1203
1204     return model;
1205
1206 }
1207
1208 //for 403 access denied page
1209 @RequestMapping(value = "/403", method = RequestMethod.GET)
1210 public ModelAndView accesssDenied() {
1211
1212     ModelAndView model = new ModelAndView();
1213
1214     //check if user is login
1215     Authentication auth = SecurityContextHolder.getContext().getAuthentication();
1216     if (!(auth instanceof AnonymousAuthenticationToken)) {
1217         UserDetails userDetail = (UserDetails) auth.getPrincipal();
1218         model.addObject("username", userDetail.getUsername());
1219     }
1220
1221     model.setViewName("403");
1222     return model;
1223
1224 }
```


1225