```
1 1. 소개
 2
     1)웹 보안 개요
 3
      -All-in-One Java 애플리케이션 개발, 전병선, 2014 참조
      -웹 어플리케이션을 작성하면서 보안과 관련해서 고려할 사항은 다음과 같다.
4
 5
      -인증(authentication): 사용자가 자신임을 증명하는 속성
 6
      -권한(authorization): 사용자가 어떤 리소스에 접근하게 할지를 결정하는 속성
 7
      -기밀성(privacy or confidentiality) : 권한이 없는 사용자에게 접근을 허용하지 않게 하는 속성
8
      -무결성(integrity): 권한이 없는 사용자가 데이터를 변경하지 못하게 하는 속성
9
      -부인방지(repudiation): 사용자가 자신이 했다는 것을 부인하지 못하게 하는 속성
10
11
     2)Spring Security는 이들 보안 요소 중에서 인증과 권한 service를 제공한다.
     3)인증은 사용자가 자신임을 증명하는 것이고, 권한은 사용자가 어떤 resource에 접근할 수 있는지를 결정하는 것이다.
12
13
     4)우리가 Web Application의 resource에 접근할 수 있는지 여부는 권한과 관련되어 있다.
14
     5)사용자가 권한이 있는지를 알기 위해서는 권한을 가진 사용자인지 인증되어야 한다.
15
     6)그러니까 먼저 사용자가 인증되어야 하고, 인증된 사용자에 부여된 권한으로 resource에 접근할 수 있는지를 check한
16
     7)사용자를 인증하는 다양한 방법이 있지만 가장 보편적으로 사용하는 방법은 form 인증 방식이다.
17
     8)사용자가 인증된 후에는 resource에 접근할 수 있는 권한을 가졌는지를 check해야 한다.
     9)사용자의 권한을 수립하는 첫 번째 단계는 principle 형식으로 사용자를 표현하는 것이다.
18
     10)principle은 resource에 접근하기 위한 보안 식별자가 할당된 계정 보유자로서, 사용자, group, service,
19
     computer 등이 principle이 될 수 있다.
20
     11)개별 사용자에 대해 일일이 권한을 가졌는지를 검사하는 것을 비효율적일 뿐만 아니라 경우에 따라서는 불가능하기도 하
21
     12)따라서, 사용자에게 권한을 부여할 때 가장 많이 사용하는 일반적인 방법은 사용자를 역할(role)에 할당하는 역할 기반
     방식이다.
     13)ROLE_ADMIN, ROLE_USER, ROLE_MNAGER 등의 역할을 정의하고, 개별 사용자에게 역할을 할당한다.
22
23
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
      -com.example.biz > Finish
30
31
32
     2)server.xml에 project 추가
33
      -Tomcat v9.0 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>4.3.24.RELEASE</org.springframework-version>
40
        <org.aspectj-version>1.9.4/org.aspectj-version>
        <org.slf4j-version>1.7.26</org.slf4j-version>
41
42
43
        <!-- Servlet -->
44
        <dependency>
45
          <groupId>javax.servlet
46
          <artifactId>iavax.servlet-api</artifactId>
47
          <version>4.0.1</version>
48
          <scope>provided</scope>
49
        </dependency>
```

<dependency>

50

```
51
            <groupId>javax.servlet.jsp</groupId>
 52
            <artifactId>javax.servlet.jsp-api</artifactId>
 53
            <version>2.3.3</version>
            <scope>provided</scope>
 54
 55
          </dependency>
 56
          <!-- Test -->
 57
 58
          <dependency>
 59
            <groupId>junit</groupId>
            <artifactId>junit</artifactId>
 60
            <version>4.12</version>
 61
 62
            <scope>test</scope>
 63
          </dependency>
 64
 65
          <dependency>
 66
            <groupId>junit</groupId>
 67
            <artifactId>junit</artifactId>
 68
            <version>4.12</version>
 69
            <scope>test</scope>
 70
          </dependency>
 71
 72
        -In mynrepository, search 'spring security'
        -Select 'Spring Security Core' 4.2.6
 73
        -Select 'Spring Security Web' 4.2.6
 74
 75
        -Select 'Spring Security Config' 4.2.6
 76
        -Select 'Spring Security Taglibs' 4.2.6
 77
        -pom.xml > right-click > Run As > Maven clean and Maven install
 78
 79
        -spring-security-core
 80
          --Spring security framework의 핵심 class와 interface를 정의한다.
 81
          --이 module은 Spring security를 사용하는 모든 application에 필수 항목이다.
 82
        -spring-security-web
 83
          --Web application의 보안을 위한 지원을 제공한다.
        -spring-security-config
 84
          --Spring security는 Spring tx 및 mvc schema와 비슷하게 Spring security의 기능을 간편하게 구성하기
 85
          위한 security schema를 제공한다.
          --security namespace의 요소를 구문분석한다.
 86
 87
        -spring-security-taglibs
 88
          --보안 정보에 접근하고 JSP page에 표시된 내용을 보호하는 tag를 정의한다.
 89
        -spring-security-acl
 90
          --ACL(접근 제어 목록)을 사용해 application에서 domain 객체의 instance를 보호할 수 있게 한다.
 91
 92
      4)Web 요청 보안 구성
 93
        -Application에서 web 요청을 보호하는 방법은 다음과 같다.
 94
          --web.xml에서 Spring의 DelegatingFilterProxy filter를 구성한다.
 95
          --Spring Security Framework에서 제공하는 요청 보안을 활성화한다.
 96
 97
      5)DelegatingFilterProxy filter 구성
        -spring-web-4.x.x.RELEASE.jar file에 들어있는 Spring Framework의 web module servlet API의
 98
        Filter interface를 구현하는 DelegatingFilterProxv를 정의한다.
 99
        -In web.xml 에 code 추가
          --반드시 filter의 이름은 springSecurityFilterChain 이어야 한다.
100
101
          <filter>
102
```

```
<filter-name>springSecurityFilterChain</filter-name>
103
104
            <filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>
          </filter>
105
106
107
          <filter-mapping>
            <filter-name>springSecurityFilterChain</filter-name>
108
109
            <url-pattern>/*</url-pattern>
110
          </filter-mapping>
111
112
        -위의 code에서 <filter-mapping>요소에는 DelegatingFilterProxy filter를 들어오는 모든 web 요청과 연결
        하도록 지정했다.
        -<filter-name> 요소에는 DelegatingFilterProxy filter가 요청을 처리하도록 위임할 Spring bean의 이름을
113
        지정한다.
114
        -이 code에서는 DelegatingFilterProxy filter가 수신하는 web 요청은 root application context의
        springSecurityFilterChain이라는 Spring bean에 위임된다.
115
116
      6)보안 관련 설정 file 만들기
        -/src/main/webapp/WEB-INF/conig folder 생성
117
118
        -project > right-click > Build Path > Config Build Path
        -Source tab > Add Folder > select src/main/webapp/WEB-INF/
119
120
        -Click [Create New Folder] > Folder name : config > OK > Apply and Close
121
        -src/main/webapp/WEB-INF/config > right-click > New > Spring Bean Configuration File
122
        -File name : securityContext.xml > Next
123
        -Check beans, security > Finish
124
        -In /WEB-INF/web.xml에 보안 관련 설정 file 등록
125
126
          <context-param>
127
            <param-name>contextConfigLocation</param-name>
128
            <param-value>classpath:securityContext.xml</param-value>
129
          </context-param>
130
131
      7)Web 요청에 대한 보안 구성
132
        -DelegatingFilterProxy filter를 구성했으니 이제 web 요청에 대한 보안을 구성하도록 하겠다.
133
        -In securityContext.xml code 추가
134
135
          <security:http auto-config="true">
136
            <security:intercept-url pattern="/login.html" access="hasRole('ROLE USER')" />
            <security:intercept-url pattern="/welcome.html" access="hasRole('ROLE ADMIN')" />
137
138
          </security:http>
139
140
        -auto-config='true'를 설정한 것만으로 기본 login page/HTTP 기본인증/logout 기능등을 제공한다.
        -만일 여기서 use-expressions='true'라고 하면 SpEL을 사용한다는 의미이다.
141
142
        -use-expressions은 기본값이 false이다.
143
        -이럴때에는 SpEL을 사용하지 않는다는 의미이다.
144
        -<intercept-url pattern="..." access="ROLE_ANONYMOUS" />
145
146
        -<intercept-url pattern="..." access="IS AUTHENTICATED ANONYMOUSLY" />
        -<intercept-url pattern="..." access="ROLE_USER" />
147
        -<intercept-url pattern="..." access="ROLE_ADMIN" />
148
149
        -<intercept-url> 요소의 access 속성은 bool 값으로 평가되는 SpEL 식이다.
150
        -해당 URL 에 접근하기위한 권한을 설정하여준다.
151
        -접근가능한 IP 등을 설정할 수도 있다.
152
        -그리고 권한은 위쪽이 우선시된다.
153
        -만일 SpEL식이 true를 반환하는 경우 사용자는 pattern과 일치하는 URL에 접근할 수 있으며 false를 반환하는 경
```

```
우 pattern 속성과 일치하는 URL에 대한 접근이 거부된다.
154
        -Spring Security Framework에는 hasRole, hasAnyRole, isAnonymous 등의 몇 가지 기본식을 제공한다.
155
        -만일 hasAnyRole('ROLE CUSTOMER', 'ROLE ADMIN')식은 사용자에게 ROLE CUSTOMER 또는
        ROLE ADMIN 역할이 있는 경우 true를 반환한다.
        -또한 pattern이 만일 /**이면 모든 URL과 일치한다는 뜻이다.
156
157
        -<intercept-url pattern="/login" access="permitAll" />
158
159
          --/login 으로는 모두 허용해준다. /login 을 막아놓으면 안되니까.
160
        -<intercept-url pattern="/resources/**" access="permitAll" />
161
162
          --Resource도 허용
163
164
        -<intercept-url pattern="/**" access="hasRole('ROLE_USER')" />
165
          --나머지는 모두 ROLE_USER 권한을 가진사람만 허용해준다.
166
167
        <security:authentication-manager>
168
          <security:authentication-provider>
169
            <security:user-service>
170
             <security:user name="javaexpert" password="12345678"</pre>
             authorities="ROLE USER"/>
171
             <security:user name="admin" password="12345678"</pre>
             authorities="ROLE ADMIN,ROLE USER"/>
172
            </security:user-service>
173
          </security:authentication-provider>
174
        </security:authentication-manager>
175
176
        -<authentication-manager> 요소는 AuthenticationManager instance를 구성하고
        <authentication-provider>는 AuthenticationProvider instance를 구성한다.
177
        -기본적으로 <authentication-provider>는 Spring UserDetailsService를 DAO로 사용해서 지정된 사용자
        이름을 참고해 사용자 저장소에서 사용자 세부 정보를 load한다.
178
        -DaoAuthenticationProvider는 사용자가 제공한 자격 증명과 구성된 UserDetailsService가 load한 사용자 세
        부 정보를 비교해 인증을 수행한다.
179
        -UserDetailsService는 데이터 원본(Data Source), 일반 file 또는 다른 사용자 저장소에서 사용자 세부 정보를
        load할 수 있다.
180
        -<authentication-provider> 하위의 <user-service>는 <user>가 정의하는 사용자를 load하는 memory
        내 UserDetailsService를 구성한다.
        -위의 예제에서는 ROLE USER와 ROLE ADMIN의 사용자를 정의했다.
181
182
        -name 속성에는 사용자에게 할당된 사용자 이름을 지정하며 password 속성에는 사용자에게 할당된 암호를 지정한다.
183
        -authorities 속성에는 사용자에게 할당된 role을 지정한다.
184
        -<authentication-provider user-service-ref="memberService"/>
185
186
          --사용자이름/비밀번호를 제공해줄 서비스 등록
187
        -위의 예제는 Inmemory로 처리하고 있음.
188
      8)web.xml 에서 코드 수정
189
190
191
        <servlet-mapping>
192
          <servlet-name>appServlet</servlet-name>
193
          <url-pattern>*.html</url-pattern> <--여기 수정할 것
194
        </servlet-mapping>
195
196
      9) Test
197
        -/WEB-INF/views/security folder 생성
198
        -/WEB-INF/views/security/login.jsp
```

```
199
200
          <body>
201
            <h1>login.jsp</h1>
202
          </body>
203
204
        -/WEB-INF/views/security/welcome.jsp
205
206
          <body>
207
            <h1>welcome.jsp</h1>
208
          </body>
209
210
        -/WEB-INF/views/home.jsp
211
212
          <@@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
          < @ page session = "false" %>
213
214
          <html>
215
          <head>
216
            <title>Home</title>
217
          </head>
218
          <body>
219
          <h1>
220
            ${greeting}
221
          </h1>
222
223
          </body>
224
          </html>
225
226
        -HomeController.java 코드 수정
227
228
          @RequestMapping(value = "/index.html", method = RequestMethod.GET)
          public String home(Model model) {
229
230
            model.addAttribute("greeting", "Hello! Spring Security");
            return "home";
231
232
          }
233
234
          @RequestMapping("/login.html")
235
          public String login(Model model) {
            return "security/login";
236
237
          }
238
239
          @RequestMapping("/welcome.html")
240
          public String welcome(Model model) {
241
            return "security/welcome";
242
          }
243
        -project > right-click > Run As > Run on Server
244
245
246
        -/login.html로 들어가면 정상적으로 login.jsp 로 이동한다.
247
        -username과 password를 미리 저장된 값을 넣지 않으면 Bad credentials로 로그인에러 발생한다.
248
        -/welcome.html로 들어가면 User와 Password를 입력할 수 있는 창으로 이동한다.
249
        -여기서 user와 password를 잘못 입력하면 'Bad credentials' 에러 메시지가 나온다.
250
        -User에 미리 설정한 javaexpert, Password에 12345678을 입력하면 403 Forbidden 오류가 나온다.
251
        -즉 권한이 없다는 뜻이다.
252
        -만일 미리 설정한 admin/12345678을 입력하면 정상적으로 welcome.jsp로 이동한다.
```

```
253
        -왜냐하면 이 계정은 ROLE ADMIN 권한을 갖고 있기 때문이다.
254
255
256 3. Login page 만들기
257
      1)securityContext.xml code 변경
258
259
        <security:http auto-config="true">
260
          <security:form-login login-page="/loginForm.html"</pre>
            username-parameter="j_username"
261
            password-parameter="j_password"
262
            login-processing-url="/j_spring_security_check" />
263
          <security:intercept-url pattern="/login.html"</pre>
264
265
            access="hasRole('ROLE_USER')" />
266
          <security:intercept-url</pre>
267
            pattern="/welcome.html" access="hasRole('ROLE_ADMIN')" />
          <security:csrf disabled="true"/>
268
269
        </security:http>
270
271
        -form 기반 인증을 사용하고자 할 때는 아래와 같이 설정한다.
272
          <form-login login-page="/login"
273
            default-target-url="/monitering"
274
            username-parameter="username"
275
            password-parameter="password"
            authentication-failure-url="/login?error"
276
            always-use-default-target='true'
277
278
          />
279
280
        -사용자가 만든 login page를 login-page 속성을 통해 Spring에게 알려준다.
281
        -이거 설정을 안하면 Spring이 기본적으로 내장된 것을 사용.
        -default-target-url="/monitering" 은 login 성공하면 이동할 page를 설정한다.
282
283
        -authentication-failure-url="/login?error" 은 login 실패시 호출해줄 URL(login page에 error
        parameter를 보내준다)
        -always-use-default-target='true' 이거 안하면 login 성공해도 /monitering 로 제대로 안갈 수 있다.
284
285
        logout invalidate-session="true" logout-url="/logout" logout-success-url="/login?logout"
286
        />
287
288
        -Logout되면 session을 초기화한다.
289
        -logout-success-url="/login?logout" 은 logout되면 이동할 page
290
        -logout-url="/logout" 은 logout을 위한 URL설정.
291
        -이거 안해주면 default로 j spring security logout 해주면됨.
292
        -<csrf/> 요소는 간단한 설정으로 csrf 를 통한 해킹을 막을 수 있다.
        ( CSRF 설명:
293
        http://tm-csc.tistory.com/entry/7-WebHacking%EC%9B%B9%ED%95%B4%ED%82%B9-CS
        RF)
294
295
296
      2)/views/security/loginForm.jsp
297
298
        <@ page language="java" contentType="text/html; charset=UTF-8"
299
          pageEncoding="UTF-8"%>
        < @ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
300
301
        <!DOCTYPE html>
        <html>
302
```

```
303
         <head>
304
         <meta charset="UTF-8">
        <title>로그인 창</title>
305
306
         </head>
307
         <body>
308
          <h1>loginForm.jsp</h1>
309
310
          <form action="<c:url value="j_spring_security_check" /> " method="post">
            ID : <input type="text" name="j_username"> <br />
311
            PW: <input type="text" name="j password"> <br />
312
313
            <input type="submit" value="LOGIN"> <br />
314
           </form>
315
         </body>
316
         </html>
317
318
      3)HomeController.java code 추가
319
        @RequestMapping("/loginForm.html")
        public String loginForm(Model model) {
320
321
          return "security/loginForm";
322
        }
323
324
      4)Test
325
        -project > right-click > Run As > Run on Server
326
        -http://localhost:8080/biz/login.html 을 요청하면 http://localhost:8080/biz/loginForm.html 로
        redirect
327
        -ID와 Password를 미리 설정한 javaexpert/12345678을 넣으면 정상적으로 login.jsp로 이동한다.
328
        -http://localhost:8080/biz/welcome.html 을 요청하면 http://localhost:8080/biz/loginForm.html
        로 redirect
329
        -만일 ID와 Password를 미리 설정한 javaexpert/12345678을 넣으면 403 forbidden error가 발생한다.
330
        -만일 ID와 Password를 미리 설정한 admin/12345678을 넣으면 정상적으로 welcome.jsp로 이동한다.
331
        -만일 잘못된 ID와 Password를 입력하면 계속 loginForm.html?error가 나타난다.
332
333
        -만일 Could not verify the provided CSRF token because your session was not found. 오류가 발생
        하면 아래의 조치를 취한다.
334
          --securityContext.xml 에서 코드 추가한다.
335
            <security:http auto-config="true">
              <security:form-login login-page="/loginForm.html" />
336
              <security:intercept-url pattern="/login.html"</pre>
337
338
                access="hasRole('ROLE_USER')" />
339
              <security:intercept-url</pre>
340
                pattern="/welcome.html" access="hasRole('ROLE ADMIN')" />
341
              <security:csrf disabled="true"/>
                                                    <---cpde 추가
342
            </security:http>
343
344
345 4. login 실패시 실패 message 출력하기
346
      1)securityContext.xml code 추가
347
348
         <security:form-login login-page="/loginForm.html"</pre>
349
          username-parameter="j username"
350
          password-parameter="j password"
          authentication-failure-url="/loginForm.html?ng"
351
                                                         <---code 추가
352
          login-processing-url="/j_spring_security_check" />
353
```

```
2)loginForm.jsp 코드 추가
354
355
356
        <h1>loginForm.jsp</h1>
357
        <c:url value="j_spring_security_check" var="loginUrl" /> <--추가
358
        <form action="${loginUrl}" method="post">
                                                             <-- 수정
359
          <c:if test="${param.ng != null}">
360
            >
361
              Login Failure < br />
362
              <c:if test="${SPRING_SECURITY_LAST_EXCEPTION != null}">
363
               Message: <c:out value="${SPRING SECURITY LAST EXCEPTION.message}"/>
364
              </c:if>
365
            366
          </c:if>
          ID : <input type="text" name="j_username"> <br />
367
          PW: <input type="text" name="j_password"> <br />
368
369
          <input type="submit" value="LOGIN"> <br />
370
        </form>
371
372
      3)Test
373
        -http://localhost:8080/biz/login.html 을 요청하면 http://localhost:8080/biz/loginForm.html 로
374
        -ID와 Password를 미리 설정한 javaexpert/12345678을 넣으면 정상적으로 login.jsp로 이동한다.
375
        -만일 잘못된 ID나 Password를 입력하면 아래와 같은 오류 message가 나온다.
376
377
          Login Failure
378
          Message: Bad credentials
379
380
        -http://localhost:8080/biz/welcome.html 을 요청하면 http://localhost:8080/biz/loginForm.html
        로 redirect
381
        -만일 ID와 Password를 미리 설정한 javaexpert/12345678을 넣으면 403 forbidden error가 발생한다.
382
        -만일 ID와 Password를 미리 설정한 admin/12345678을 넣으면 정상적으로 welcome.jsp로 이동한다.
383
        -만일 잘못된 ID와 Password를 입력하면 계속 loginForm.html?ng가 나오면서 아래의 message가 나타난다.
384
385
          Login Failure
386
          Message: Bad credentials
387
388
389 5. login, logout 상태 표시하기
390
      1)security/login.jsp page code 추가
391
392
          <@@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
393
          <body>
394
            <h1>login.jsp</h1>
395
396
            <c:if test="${not empty pageContext.request.userPrincipal}">
397
              현재 로그인 상태
398
            </c:if>
399
            <c:if test="${empty pageContext.request.userPrincipal}">
400
              현재 로그아웃 상태
            </c:if>
401
402
403
            User ID: ${pageContext.request.userPrincipal} < br />
404
            <a href="${pageContext.request.contextPath}/">Log Out</a>
405
          </body>
```

```
406
407
      2)Test
408
        -http://localhost:8080/biz/login.html 을 요청하면 http://localhost:8080/biz/loginForm.html 로
409
        -아이디와 패스워드를 미리 설정한 javaexpert/12345678을 넣으면 정상적으로 login.jsp로 이동한다.
410
        -이 때 화면에는 아래의 message가 나온다.
411
412
          현재 로그인 상태
413
          User ID:
          org.springframework.security.authentication.UsernamePasswordAuthenticationToken@b09
          a9234:
414
          Principal: org.springframework.security.core.userdetails.User@f486f6cc:
415
          Username: javaexpert;
          Password: [PROTECTED];
416
          Enabled: true;
417
418
          AccountNonExpired: true:
419
          credentialsNonExpired: true;
420
          AccountNonLocked: true;
421
          Granted Authorities: ROLE USER:
422
          Credentials: [PROTECTED];
423
          Authenticated: true;
424
          Details:
          org.springframework.security.web.authentication.WebAuthenticationDetails@166c8:
425
          RemoteIpAddress: 0:0:0:0:0:0:0:1;
426
          SessionId: 949A2731FFC91E3C350818E0772C8E45;
427
          Granted Authorities: ROLE USER
428
429
          Log Out
430
431
      3)여기서 username만 출력하려면
432
433
        User ID: ${pageContext.request.userPrincipal.name} 으로 변경한다.
434
435
      4)securityContext.xml code 추가
436
437
          <security:form-login login-page="/loginForm.html"</pre>
            username-parameter="j username"
438
439
            password-parameter="j password"
            authentication-failure-url="/loginForm.html?ng"
440
441
            login-processing-url="/j_spring_security_check" />
442
          <security:logout logout-url="/j spring security logout"/>
                                                                   <--code 추가
443
444
      5)Test
445
        -로그인 창에서 Log Out link를 클릭하면 index.html로 이동한다.
446
447
448 6. Lab
449
      1)Spring Legacy Project 생성
450
        -In Package Explorer > right-click > New > Spring Legacy Project
451
        -Project name: SpringSecurityDemo1
452
        -Select Spring MVC Project > Next
453
        -com.example.biz > Finish
454
455
      2)server.xml에 project 추가
```

6/22/2019 12:09 AM

```
456
        -Tomcat v9.0 Server at localhost > right-click > Add and Remove
457
        -SpringSecurityDemo1 > Add > Finish > OK
458
459
      3)Spring security library download and install
460
        -In pom.xml 아래와 같이 수정
461
          <java-version>1.8</java-version>
          <org.springframework-version>4.3.24.RELEASE</org.springframework-version>
462
463
          <org.aspectj-version>1.9.4</org.aspectj-version>
464
          <org.slf4j-version>1.7.26</org.slf4j-version>
465
466
        -version 수정
467
          <dependency>
468
            <groupId>javax.servlet</groupId>
469
            <artifactId>javax.servlet-api</artifactId>
470
            <version>4.0.1</version>
471
            <scope>provided</scope>
472
          </dependency>
473
          <dependency>
474
            <groupId>javax.servlet.jsp</groupId>
475
            <artifactId>javax.servlet.jsp-api</artifactId>
            <version>2.3.3</version>
476
477
            <scope>provided</scope>
478
          </dependency>
479
          <dependency>
            <groupId>junit</groupId>
480
481
            <artifactId>junit</artifactId>
482
            <version>4.12</version>
483
            <scope>test</scope>
484
          </dependency>
485
486
        -In mvnrepository, search 'spring security'
487
          -Select 'Spring Security Core' 4.2.12
          -Select 'Spring Security Web' 4.2.12
488
          -Select 'Spring Security Config' 4.2.12
489
490
          -Select 'Spring Security Taglibs' 4.2.12
491
492
          <dependency>
493
            <groupId>org.springframework.security</groupId>
494
            <artifactId>spring-security-core</artifactId>
495
            <version>4.2.12.RELEASE
496
          </dependency>
497
          <dependency>
498
            <groupId>org.springframework.security</groupId>
499
            <artifactId>spring-security-web</artifactId>
            <version>4.2.12.RELEASE
500
501
          </dependency>
502
          <dependency>
503
            <groupId>org.springframework.security</groupId>
504
            <artifactId>spring-security-config</artifactId>
            <version>4.2.12.RELEASE
505
506
          </dependency>
507
          <dependency>
508
            <groupId>org.springframework.security</groupId>
            <artifactId>spring-security-taglibs</artifactId>
509
```

```
<version>4.2.12.RELEASE
510
511
          </dependency>
512
        -pom.xml > right-click > Run As > Maven clean and Maven install
513
514
        -project > right-click > Properties > Project Facets > Java > Select 1.8
515
        -Runtimes Tab > Check Apache Tomcat v9.0
516
        -Click Apply and Close
517
518
      4)HomeController.java 코드 수정
519
520
        package com.example.biz;
521
522
        import java.util.Date;
523
524
        import org.springframework.stereotype.Controller;
525
        import org.springframework.ui.Model;
526
        import org.springframework.web.bind.annotation.RequestMapping;
        import org.springframework.web.bind.annotation.RequestMethod;
527
528
529
        @Controller
530
        public class HomeController {
531
532
          @RequestMapping(value = "/", method = RequestMethod.GET)
          public String home(Model model) {
533
            model.addAttribute("serverTime", new Date());
534
535
            return "home";
536
          }
537
        }
538
539
      5)project > right-click > Run As > Run on Server
540
      6)src/main/webapp/WEB-INF/spring/Spring Bean Configuration File > spring-security.xml 파일
541
      생성
542
        -check 'security'
543
544
        <security:http auto-config="true" use-expressions="true">
545
         </security:http>
546
547
         <security:authentication-manager>
548
         </security:authentication-manager>
549
550
      7)web.xml 코드 수정
551
552
         <context-param>
553
          <param-name>contextConfigLocation</param-name>
554
           <param-value>/WEB-INF/spring/spring-security.xml</param-value>
555
         </context-param>
556
557
558
        <filter>
559
          <filter-name>springSecurityFilterChain</filter-name>
560
           <filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>
561
         </filter>
562
```

```
563
         <filter-mapping>
564
           <filter-name>springSecurityFilterChain</filter-name>
565
           <url-pattern>/*</url-pattern>
         </filter-mapping>
566
567
568
      8)WEB-INF/spring/appServlet/servlet-context.xml
569
570
         <context:component-scan base-package="com.example" />
571
572
      9)project > right-click > Run As > Run on Server
573
574
      10)spring-security.xml
575
576
         <?xml version="1.0" encoding="UTF-8"?>
577
         <beans xmlns="http://www.springframework.org/schema/beans"</pre>
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
578
579
           xmlns:security="http://www.springframework.org/schema/security"
           xsi:schemaLocation="http://www.springframework.org/schema/security
580
           http://www.springframework.org/schema/security/spring-security-4.2.xsd
             http://www.springframework.org/schema/beans
581
             http://www.springframework.org/schema/beans/spring-beans.xsd">
582
583
           <security:http auto-config="true" use-expressions="true">
584
             <security:intercept-url pattern="/login"
               access="permitAII" />
585
             <security:intercept-url pattern="resources/**"</pre>
586
587
               access="permitAII" />
             <security:intercept-url pattern="/**"
588
               access="hasRole('ROLE USER')" />
589
             <security:form-login login-page="/login"</pre>
590
591
               default-target-url="/welcome" username-parameter="j_username"
592
               password-parameter="j password"
593
               authentication-failure-url="/login?error"
               login-processing-url="/j_spring_security_check"
594
               always-use-default-target="true" />
595
596
             <security:logout invalidate-session="true"</pre>
               logout-url="/logout" logout-success-url="/login?logout" />
597
598
             <security:csrf />
599
           </security:http>
600
601
           <security:authentication-manager>
             <!-- <security:authentication-provider>
602
               <security:user-service>
603
                 <security:user name="javaexpert" password="12345678"</pre>
604
                   authorities="ROLE_USER" />
605
                 <security:user name="admin" password="12345678"
606
607
                   authorities="ROLE ADMIN, ROLE USER" />
608
               </security:user-service>
             </security:authentication-provider> -->
609
610
             <security:authentication-provider user-service-ref="memberService"/>
611
           </security:authentication-manager>
612
613
           <bean id="memberService"</pre>
614
             class="com.example.service.MemberService">
```

```
cproperty name="userService" ref="userService" />
615
616
          </bean>
          <bean id="userService"</pre>
617
618
            class="com.example.service.UserService">
            cproperty name="userDao" ref="userDao" />
619
620
           </bean>
621
           <bean id="userDao" class="com.example.dao.UserDao">
            roperty name="username" value="javaexpert" />
622
            cproperty name="password" value="12345678" />
623
624
           </bean>
625
         </beans>
626
627
      9)HomeController.java
628
629
        package com.example.biz;
630
631
        import java.util.Date;
632
633
        import org.springframework.stereotype.Controller;
634
        import org.springframework.ui.Model;
635
        import org.springframework.web.bind.annotation.RequestMapping;
636
        import org.springframework.web.bind.annotation.RequestMethod;
        import org.springframework.web.bind.annotation.RequestParam;
637
638
        import org.springframework.web.servlet.ModelAndView;
639
640
        @Controller
641
        public class HomeController {
642
643
          @RequestMapping(value = "/", method = RequestMethod.GET)
644
          public String home(Model model) {
645
            model.addAttribute("serverTime", new Date());
646
            return "home";
647
648
649
          @RequestMapping(value = "/login", method = RequestMethod.GET)
650
          public ModelAndView login(
            @RequestParam(value = "error", required = false) String error,
651
            @RequestParam(value = "logout", required = false) String logout) {
652
653
            ModelAndView model = new ModelAndView();
654
            if (error != null) {
              model.addObject("error", "Invalid username and password!");
655
656
657
            if (logout != null) {
658
              model.addObject("msg", "You've been logged out successfully.");
659
660
            model.setViewName("login");
661
            return model;
662
          }
663
          @RequestMapping("/welcome")
664
665
          public String welcome(Model model) {
            return "welcome";
666
667
668
```

```
@RequestMapping(value = "/logout", method = RequestMethod.GET)
669
670
          public String logout(Model model) {
            return "logout";
671
672
673
        }
674
675
676
      10)com.example.vo.UserVO.java
677
678
        package com.example.vo;
679
680
        public class UserVO {
681
          private String username;
682
          private String password;
683
684
          public UserVO() {}
685
686
          public UserVO(String username, String password) {
687
            this.username = username;
688
            this.password = password;
689
690
691
          public String getUsername() {
692
            return username;
693
          }
694
695
          public void setUsername(String username) {
696
            this.username = username;
697
698
699
          public String getPassword() {
700
            return password;
701
702
703
          public void setPassword(String password) {
704
            this.password = password;
705
          }
706
707
          @Override
708
          public String toString() {
            return "UserVO [username=" + username + ", password=" + password + "]";
709
710
711
        }
712
713
      11)com.example.dao.UserDao.java
714
715
        package com.example.dao;
716
717
        import org.springframework.stereotype.Repository;
718
719
        import com.example.vo.UserVO;
720
721
        @Repository("userDao")
722
        public class UserDao {
```

```
723
          private String username;
724
          private String password;
725
726
          public void setUsername(String username) {
727
            this.username = username;
728
          }
729
730
          public void setPassword(String password) {
731
            this.password = password;
732
733
734
          public UserVO getUsersByID(String username) {
             return new UserVO(this.username, this.password);
735
736
737
         }
738
739
       12)com.example.service.UserService.java
740
741
         package com.example.service;
742
743
         import org.springframework.beans.factory.annotation.Autowired;
744
         import org.springframework.stereotype.Service;
745
746
         import com.example.dao.UserDao;
747
         import com.example.vo.UserVO;
748
749
         @Service("userService")
750
         public class UserService {
751
          private UserDao userDao;
752
753
          public void setUserDao(UserDao userDao) {
754
            this.userDao = userDao;
755
756
          public UserVO getUsersByID(String username) {
757
758
            return this.userDao.getUsersByID(username);
759
          }
         }
760
761
762
       13)com.example.service.MemberService.java
763
764
         package com.example.service;
765
766
         import java.util.ArrayList;
767
         import java.util.Collection;
768
769
         import org.springframework.beans.factory.annotation.Autowired;
770
         import org.springframework.security.core.authority.SimpleGrantedAuthority;
771
         import org.springframework.security.core.userdetails.User;
772
         import org.springframework.security.core.userdetails.UserDetails;
773
         import org.springframework.security.core.userdetails.UserDetailsService;
774
         import org.springframework.security.core.userdetails.UsernameNotFoundException;
775
         import org.springframework.stereotype.Service;
776
```

```
777
        import com.example.vo.UserVO;
778
779
        @Service("memberService")
780
        public class MemberService implements UserDetailsService{
781
          private UserService userService;
782
783
784
           * 사용자가 입력한 ID/PWD 를 검증하기위해 저장소에 그 ID/PWD 가 있는지 확인하는 source이다. 저장소는
785
           in-memory가 될 수도있고,
786
           * DB가 될 수도있고 어떠한 것(Facebook,Naver등등)도 될 수있다. userService는 DB 에 접근하기 위한
           service이다.
787
           */
788
789
          public void setUserService(UserService userService) {
790
            this.userService = userService:
791
792
793
          @Override
794
          public UserDetails loadUserByUsername(String username) throws
          UsernameNotFoundException {
795
            UserVO userVO = this.userService.getUsersByID(username);
796
            if(userVO == null)
              throw new UsernameNotFoundException("No user found with username " +
797
              userVO.getUsername());
798
            Collection < SimpleGrantedAuthority > roles = new ArrayList < SimpleGrantedAuthority > ();
            roles.add(new SimpleGrantedAuthority("ROLE_USER"));
799
800
            UserDetails user = new User(username, userVO.getPassword(), roles);
801
            return user:
802
          }
803
        }
804
      14)views/login.jsp
805
806
807
        <%@ page language="java" contentType="text/html; charset=UTF-8"</pre>
808
          pageEncoding="UTF-8"%>
        <@@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
809
        <!DOCTYPE html>
810
811
        <html>
812
        <head>
813
          <title>로그인 페이지</title>
814
          <meta charset="UTF-8">
815
          <script>
816
            function doLogin() {
817
              if(frm.j_username.value == "") {
                alert("아이디를 입력해주세요.");
818
819
                return;
820
821
              if(frm.j_password.value == "") {
                alert("패스워드를 입력해주세요.");
822
823
                return:
824
825
              frm.submit();
826
            }
```

```
827
          </script>
828
        </head>
829
        <body>
          <form name="frm" action="j spring security check" method="post">
830
831
              >
832
833
                <label for="userID">ID</label>
834
                <input id = "userID" type="userID" name="j_username" placeholder="ID"</pre>
                required>
835
              836
              <
837
                <label for="password">Password</label>
838
                <input id = "password" type="password" name="j_password"
                placeholder="PASSWORD" required>
839
              >
                <input type="button" value="로그인" onclick="doLogin()"/>
840
841
              842
            <input type="hidden" name="${ csrf.parameterName}"</pre>
843
844
              value="${_csrf.token}" />
845
          </form>
846
          <c:if test="${not empty error}">
847
848
            <div class="error">${error}</div>
849
          </c:if>
850
          <c:if test="${not empty msg}">
851
            <div class="msg">${msg}</div>
852
          </c:if>
853
        </body>
854
        </html>
855
856
      15) views/welcome.jsp
857
858
        <@ page language="java" contentType="text/html; charset=UTF-8"
859
          pageEncoding="UTF-8"%>
860
        <@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
        <!DOCTYPE html>
861
        <html>
862
863
        <head>
864
        <meta charset="UTF-8">
865
        <title>Member Page</title>
866
        </head>
867
        <body>
868
          <c:if test="${pageContext.request.userPrincipal.name != null}">
869
870
              Welcome: ${pageContext.reguest.userPrincipal.name} |
871
              <a href="logout"> Logout</a>
872
            </h2>
873
          </c:if>
874
        </body>
875
        </html>
876
877
      16)logout.jsp
878
```

```
<%@ page language="java" contentType="text/html; charset=UTF-8"</pre>
879
880
          pageEncoding="UTF-8"%>
        <@@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
881
882
        <c:set var="username" value="${pageContext.request.userPrincipal.name}" />
883
        <!DOCTYPE html>
884
        <html>
885
        <head>
        <meta charset="UTF-8">
886
        <title>Logout page</title>
887
888
        </head>
889
        <body>
          <c:url value="/logout" var="logoutUrl" />
890
891
          <!-- csrt for log out-->
892
          <form action="${logoutUrl}" method="post" id="logoutForm">
893
           <input type="hidden" name="${_csrf.parameterName}"</pre>
894
            value="${ csrf.token}" />
895
          </form>
896
          <script>
897
            function formSubmit() {
898
              document.getElementById("logoutForm").submit();
899
          </script>
900
901
          <form>
902
            <span style="color: gray;" ><h5> ${username} 님 반갑습니다.
            <a href = "javascript:formSubmit()"> 로그아웃 </a> </h5></span>
903
904
          </form>
905
        </body>
906
        </html>
907
908
      17)Test
909
        -SpringSecurityDemo1 > right-click > Run As > Run on Server
910
        -http://localhost:8080/biz/login 으로 이동되며
911
        -이때 ID와 Password를 잘 못 입력하면 Invalid username and password! 를 보며
        -javaexpert/12345678을 입력하면 welcome으로 이동한다.
912
913
        -welcome에서 Logout link click하면
914
        -logout으로 이동하고 [로그아웃] link를 click하면 다시 첫 page로 이동하지만, You've been logged out
        successfully, message를 만난다.
915
916
917 7. 보안관련 taglibs 사용하기
918
      1)SpringSecurityDemo project에서
      2) pom.xml 에서 spring-security-taglibs : 4.2.12.RELEASE 확인
919
920
      3)security/login.jsp에서
921
        -아래 taglib 추가
          < @ taglib uri="http://www.springframework.org/security/tags" prefix="security" %>
922
923
924
        -코드 변경
925
926
          <c:if test="${not empty pageContext.request.userPrincipal}">
927
            928
          </c:if>
          <c:if test="${empty pageContext.request.userPrincipal}">
929
930
            현재 로그아웃 상태
931
          </c:if>
```

```
932
933
        -위의 코드를 아래의 코드로 변경
934
935
          <security:authorize access="hasRole('ROLE_USER')">
936
            현재 로그인 상태
937
          </security:authorize>
938
          <security:authorize access="hasRole('ROLE ANONYMOUS')">
939
            현재 로그아웃 상태
940
          </security:authorize>
941
942
        -코드 변경
943
944
          User ID: ${pageContext.request.userPrincipal.name}<br/>br/>
945
946
        -아래의 코드로 변경
947
948
          User ID: <security:authentication property="name"/><br/>
949
950
      4)Test
951
        -http://localhost:8080/biz/login.html 을 요청하면 http://localhost:8080/biz/loginForm.html 로
952
        -아이디와 패스워드를 미리 설정한 javaexpert/12345678을 넣으면 정상적으로 login.jsp로 이동한다.
953
        -이 때 화면에는 아래의 message가 나온다.
954
955
          login.jsp
956
          현재 로그인 상태
957
          User ID: javaexpert
958
          Log Out
959
960
        -Logout link를 click하면 다시 첫 page로 이동한다.
961
962
963 8. Database-based Login Authentication
964
      1)Spring Legacy Project 생성
965
        -In Package Explorer > right-click > New > Spring Legacy Project
966
        -Project name: SpringSecurityDatabaseDemo
        -Select Spring MVC Project > Next
967
        -com.example.biz > Finish
968
969
970
      2)server.xml에 project 추가
971
        -Tomcat v9.0 Server at localhost > right-click > Add and Remove
972
        -SpringSecurityDatabaseDemo > Add > Finish > OK
973
974
      3)Spring security library download and install
975
        -In pom.xml 아래와 같이 수정
976
          <java-version>1.8</java-version>
977
          <org.springframework-version>4.3.24.RELEASE</org.springframework-version>
978
          <org.aspectj-version>1.9.4</org.aspectj-version>
979
          <org.slf4j-version>1.7.26</org.slf4j-version>
980
981
        -version 수정
982
          <dependency>
983
            <groupId>javax.servlet
984
            <artifactId>javax.servlet-api</artifactId>
```

```
985
             <version>4.0.1</version>
 986
             <scope>provided</scope>
            </dependency>
 987
 988
            <dependency>
 989
             <groupId>javax.servlet.jsp</groupId>
 990
             <artifactId>javax.servlet.jsp-api</artifactId>
 991
             <version>2.3.3</version>
 992
             <scope>provided</scope>
 993
           </dependency>
 994
           <dependency>
 995
             <groupId>junit</groupId>
             <artifactId>junit</artifactId>
 996
 997
             <version>4.12</version>
 998
             <scope>test</scope>
 999
            </dependency>
1000
1001
         -In mvnrepository, search 'spring security'
           -Select 'Spring Security Core' 4.2.12
1002
           -Select 'Spring Security Web' 4.2.12
1003
           -Select 'Spring Security Config' 4.2.12
1004
           -Select 'Spring Security Taglibs' 4.2.12
1005
1006
1007
           <dependency>
1008
             <groupId>org.springframework.security</groupId>
             <artifactId>spring-security-core</artifactId>
1009
1010
             <version>4.2.12.RELEASE
           </dependency>
1011
1012
           <dependency>
1013
             <groupId>org.springframework.security</groupId>
             <artifactId>spring-security-web</artifactId>
1014
1015
             <version>4.2.12.RELEASE
1016
            </dependency>
1017
           <dependency>
             <groupId>org.springframework.security</groupId>
1018
1019
             <artifactId>spring-security-config</artifactId>
1020
             <version>4.2.12.RELEASE
            </dependency>
1021
1022
            <dependency>
1023
             <groupId>org.springframework.security</groupId>
1024
             <artifactId>spring-security-taglibs</artifactId>
1025
             <version>4.2.12.RELEASE
1026
           </dependency>
1027
1028
         -pom.xml > right-click > Run As > Maven clean and Maven install
         -project > right-click > Properties > Project Facets > Java > Select 1.8
1029
1030
         -Runtimes Tab > Check Apache Tomcat v9.0
1031
         -Click Apply and Close
1032
1033
       4)HomeController.java
1034
1035
         package com.example.biz;
1036
1037
         import org.springframework.stereotype.Controller;
         import org.springframework.ui.Model;
1038
```

```
1039
         import org.springframework.web.bind.annotation.RequestMapping;
1040
         import org.springframework.web.bind.annotation.RequestMethod;
1041
1042
         @Controller
1043
         public class HomeController {
           @RequestMapping(value = "/", method = RequestMethod.GET)
1044
           public String home(Model model) {
1045
             model.addAttribute("greeting", "Hello Spring Security");
1046
1047
             return "home";
1048
           }
1049
         }
1050
1051
       5)home.jsp
1052
1053
          <@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
          < @ page session="false" %>
1054
1055
         <html>
1056
          <head>
1057
           <title>Home</title>
1058
          </head>
1059
          <body>
          <h1>${greeting} </h1>
1060
1061
1062
         </body>
1063
          </html>
1064
1065
       6)project > right-click > Run As > Run on Server
1066
1067
       7) Database table & sequence creation
1068
1069
         CREATE SEQUENCE seq_user_role_id
1070
         MAXVALUE 9999;
1071
1072
1073
         CREATE TABLE Users (
1074
           username VARCHAR2(20),
1075
           password VARCHAR2(20) NOT NULL,
           enabled NUMBER(1) DEFAULT 1,
1076
1077
           CONSTRAINT users_username_pk PRIMARY KEY (username)
1078
         );
1079
1080
         CREATE TABLE User_roles (
1081
           user_role_id NUMBER(4),
1082
           username VARCHAR2(45),
1083
           role VARCHAR2(45) NOT NULL,
           CONSTRAINT user roles id PRIMARY KEY (user role id),
1084
1085
           CONSTRAINT user roles uk UNIQUE (role, username),
1086
           CONSTRAINT user_roles_fk FOREIGN KEY (username) REFERENCES users (username)
1087
         );
1088
1089
       8) Inserts some records for testing.
1090
         INSERT INTO Users(username, password, enabled) VALUES ('javaexpert', '12345678', 1);
1091
         INSERT INTO users(username, password, enabled) VALUES ('alex', '12345678', 1);
1092
```

```
1093
1094
          INSERT INTO user roles (user role id, username, role)
          VALUES (seg user role id.NEXTVAL, 'javaexpert', 'ROLE USER');
1095
1096
         INSERT INTO user_roles (user_role_id, username, role)
1097
1098
         VALUES (seg user role id.NEXTVAL, 'javaexpert', 'ROLE ADMIN');
1099
1100
          COMMIT;
1101
        9)pom.xml에 oracle driver와 MyBatis 관련 dependency 추가
1102
1103
1104
          <dependency>
1105
            <groupId>com.oracle</groupId>
1106
            <artifactId>ojdbc8</artifactId>
1107
            <version>12.2</version>
1108
          </dependency>
1109
          <!-- https://mvnrepository.com/artifact/org.mybatis/mybatis -->
1110
          <dependency>
1111
            <groupId>org.mybatis</groupId>
            <artifactId>mybatis</artifactId>
1112
            <version>3.5.1</version>
1113
1114
          </dependency>
          <!-- https://mvnrepository.com/artifact/org.mybatis/mybatis-spring -->
1115
1116
          <dependency>
            <groupId>org.mybatis</groupId>
1117
1118
            <artifactId>mybatis-spring</artifactId>
1119
            <version>2.0.1</version>
1120
          </dependency>
1121
1122
        10) Database 연결을 위한 환경설정
1123
          -Build path에 config foler 추가
1124
           --project right-click > Build Path > Configure Build Path > Select [Source] tab
1125
           --Click [Add Folder] > Select 현재 project > Click [Create New Folder...]
           --Folder name : config > Finish > OK > Apply and Close
1126
           --Java Resources > config folder 확인
1127
1128
1129
          -config/dbinfo.properties file 생성
1130
1131
           db.driver=oracle.jdbc.driver.OracleDriver
1132
           db.url=idbc:oracle:thin:@localhost:1521:XE
1133
           db.username=hr
1134
           db.password=hr
1135
1136
          -pom.xml에 spring-jdbc 추가
1137
1138
            <!-- https://mvnrepository.com/artifact/org.springframework/spring-idbc -->
1139
            <dependency>
1140
              <groupId>org.springframework</groupId>
              <artifactId>spring-jdbc</artifactId>
1141
1142
              <version>4.3.24.RELEASE
1143
            </dependency>
1144
1145
          -web.xml 코드 추가
1146
```

6/22/2019 12:09 AM

```
1147
            <filter>
1148
              <filter-name>springSecurityFilterChain</filter-name>
1149
              <filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>
            </filter>
1150
1151
1152
            <filter-mapping>
1153
              <filter-name>springSecurityFilterChain</filter-name>
1154
              <url-pattern>/*</url-pattern>
1155
            </filter-mapping>
1156
1157
          -web.xml 수정
1158
1159
            <context-param>
1160
              <param-name>contextConfigLocation</param-name>
1161
              <param-value>/WEB-INF/spring/spring-security.xml</param-value>
1162
            </context-param>
1163
1164
          -src/main/webapp/WEB-INF/spring/spring-security.xml 생성
            --src/main/webapp/WEB-INF/spring > right-click > New > Spring Bean Configuration File
1165
1166
            --File name : spring-security.xml > Next
            -- Check beans, security, context > Finish
1167
1168
              <?xml version="1.0" encoding="UTF-8"?>
1169
1170
              <beans xmlns="http://www.springframework.org/schema/beans"</pre>
                xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
1171
1172
                xmlns:security="http://www.springframework.org/schema/security"
                xmlns:context="http://www.springframework.org/schema/context"
1173
                xsi:schemaLocation="http://www.springframework.org/schema/security
1174
               http://www.springframework.org/schema/security/spring-security-4.2.xsd
                 http://www.springframework.org/schema/beans
1175
                 http://www.springframework.org/schema/beans/spring-beans.xsd">
1176
                <context:property-placeholder location="classpath:dbinfo.properties"/>
1177
                <bean id="dataSource"</pre>
1178
1179
                 class="org.springframework.jdbc.datasource.DriverManagerDataSource">
1180
                  cproperty name="driverClassName"
                   value="${db.driverClass}" />
1181
                  cproperty name="url"
1182
                   value="${db.url}"/>
1183
1184
                  cproperty name="username" value="${db.username}" />
                  cproperty name="password" value="${db.password}" />
1185
1186
                </bean>
1187
                <!-- enable use-expressions -->
1188
1189
                <security:http>
1190
1191
                  <security:intercept-url pattern="/admin**" access="hasRole('ROLE_ADMIN')" />
1192
1193
                  <!-- access denied page -->
                  <security:access-denied-handler error-page="/403" />
1194
1195
1196
                  <security:form-login
1197
                   login-page="/login"
                   default-target-url="/welcome"
1198
```

```
1199
                    authentication-failure-url="/login?error"
1200
                    login-processing-url="/j_spring_security_check"
                    username-parameter="username"
1201
1202
                    password-parameter="password" />
1203
                  <security:logout invalidate-session="true"</pre>
1204
                    logout-url="/j spring security logout" logout-success-url="/login?logout" />
                  <!-- enable csrf protection -->
1205
1206
                  <security:csrf disabled="true"/>
1207
                </security:http>
1208
1209
                <!-- Select users and user roles from database -->
                <security:authentication-manager>
1210
1211
                 <security:authentication-provider>
1212
                  <security:jdbc-user-service data-source-ref="dataSource"</pre>
1213
                   users-by-username-query=
1214
                    "select username, password, enabled from users where username=?"
1215
                   authorities-by-username-query=
                    "select username, role from user_roles where username =? "/>
1216
1217
                 </security:authentication-provider>
1218
                </security:authentication-manager>
1219
              </beans>
1220
1221
        11) views/hello.jsp
1222
1223
          <@ page language="java" contentType="text/html; charset=UTF-8"
1224
            pageEncoding="UTF-8"%>
1225
          <@@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
1226
          <%@taglib prefix="sec"</pre>
            uri="http://www.springframework.org/security/tags"%>
1227
1228
          <html>
1229
          <body>
1230
            <h1>Title: ${title}</h1>
            <h1>Message: ${message}</h1>
1231
1232
1233
            <sec:authorize access="hasRole('ROLE USER')">
1234
              <!-- For login user -->
1235
              <c:url value="/j spring security logout" var="logoutUrl" />
              <form action="${logoutUrl}" method="post" id="logoutForm">
1236
                <input type="hidden" name="${_csrf.parameterName}"</pre>
1237
1238
                  value="${_csrf.token}" />
1239
              </form>
1240
              <script>
1241
                function formSubmit() {
1242
                  document.getElementById("logoutForm").submit();
1243
                }
1244
              </script>
1245
1246
              <c:if test="${pageContext.request.userPrincipal.name != null}">
1247
                <h2>
1248
                  User: <sec:authentication property="name"/> | <a
1249
                    href="javascript:formSubmit()"> Logout</a>
1250
                </h2>
1251
              </c:if>
1252
```

```
1253
1254
            </sec:authorize>
1255
          </body>
1256
          </html>
1257
1258
        12) views/login.jsp
1259
1260
          <@ page language="java" contentType="text/html; charset=UTF-8"
            pageEncoding="UTF-8"%>
1261
            <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
1262
1263
          <html>
1264
          <head>
1265
          <title>Login Page</title>
1266
          <style>
            .error {
1267
1268
              padding: 15px;
1269
              margin-bottom: 20px;
1270
              border: 1px solid transparent;
              border-radius: 4px;
1271
              color: #a94442;
1272
              background-color: #f2dede;
1273
1274
             border-color: #ebccd1;
1275
            }
1276
1277
            .msg {
1278
              padding: 15px;
              margin-bottom: 20px;
1279
1280
              border: 1px solid transparent;
1281
              border-radius: 4px;
1282
              color: #31708f;
1283
              background-color: #d9edf7;
1284
              border-color: #bce8f1;
1285
1286
            #login-box {
1287
1288
              width: 300px;
              padding: 20px;
1289
1290
              margin: 100px auto;
1291
              background: #fff;
1292
              -webkit-border-radius: 2px;
              -moz-border-radius: 2px;
1293
              border: 1px solid #000;
1294
1295
1296
          </style>
1297
          </head>
          <body onload='document.loginForm.username.focus();'>
1298
1299
1300
            <h1>Spring Security Login Form (Database Authentication)</h1>
1301
            <div id="login-box">
1302
1303
1304
              <h2>Login with Username and Password</h2>
1305
              <c:if test="${not empty error}">
1306
```

```
<div class="error">${error}</div>
1307
1308
             </c:if>
             <c:if test="${not empty msg}">
1309
1310
              <div class="msg">${msg}</div>
1311
             </c:if>
1312
1313
             <form name='loginForm'
              action="<c:url value='/j_spring_security_check' /> " method='POST'>
1314
1315
1316
             1317
              1318
                User:
1319
                <input type='text' name='username'>
1320
              1321
              1322
                Password:
1323
                <input type='password' name='password' />
1324
              1325
              1326
                <input name="submit" type="submit"
                 value="submit" />
1327
1328
              1329
              1330
              <input type="hidden" name="${ csrf.parameterName}"</pre>
1331
1332
              value="${_csrf.token}" />
1333
1334
             </form>
1335
           </div>
1336
         </body>
1337
         </html>
1338
1339
       13) views/admin.jsp
1340
1341
         <%@ page language="java" contentType="text/html; charset=UTF-8"</pre>
1342
           pageEncoding="UTF-8"%>
         <%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
1343
         <html>
1344
1345
         <body>
1346
           <h1>Title: ${title}</h1>
1347
           <h1>Message: ${message}</h1>
1348
           <c:url value="/j_spring_security_logout" var="logoutUrl" />
1349
           <form action="${logoutUrl}" method="post" id="logoutForm">
1350
             <input type="hidden" name="${_csrf.parameterName}"</pre>
1351
              value="${_csrf.token}"/>
1352
1353
           </form>
1354
           <script>
1355
            function formSubmit() {
              document.getElementById("logoutForm").submit();
1356
1357
1358
           </script>
1359
1360
           <c:if test="${pageContext.request.userPrincipal.name != null}">
```

```
<h2>
1361
1362
               Welcome: ${pageContext.reguest.userPrincipal.name} | <a
                 href="javascript:formSubmit()"> Logout</a>
1363
1364
             </h2>
1365
           </c:if>
1366
1367
          </body>
          </html>
1368
1369
1370
       14)views/403.jsp
1371
          <@ page language="java" contentType="text/html; charset=UTF-8"
1372
1373
           pageEncoding="UTF-8"%>
1374
          <@@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
1375
         <html>
1376
         <body>
1377
           <h1>HTTP Status 403 - Access is denied</h1>
1378
1379
           <c:choose>
1380
             <c:when test="${empty username}">
               <h2>You do not have permission to access this page!</h2>
1381
1382
             </c:when>
1383
             <c:otherwise>
1384
               <h2>Username: ${username} <br/>
                   You do not have permission to access this page!</h2>
1385
1386
             </c:otherwise>
           </c:choose>
1387
1388
1389
         </body>
1390
          </html>
1391
1392
       15)HomeController.java
1393
         @RequestMapping(value = { "/", "/welcome**" }, method = RequestMethod.GET)
1394
1395
         public ModelAndView defaultPage() {
1396
1397
           ModelAndView model = new ModelAndView();
           model.addObject("title", "Spring Security Login Form - Database Authentication");
1398
           model.addObject("message", "This is default page!");
1399
1400
           model.setViewName("hello");
           return model;
1401
1402
1403
         }
1404
         @RequestMapping(value = "/admin**", method = RequestMethod.GET)
1405
         public ModelAndView adminPage() {
1406
1407
1408
           ModelAndView model = new ModelAndView();
           model.addObject("title", "Spring Security Login Form - Database Authentication");
1409
           model.addObject("message", "This page is for ROLE ADMIN only!");
1410
1411
           model.setViewName("admin");
           return model;
1412
1413
1414
         }
```

```
1415
1416
          @RequestMapping(value = "/login", method = RequestMethod.GET)
          public ModelAndView login(@RequestParam(value = "error", required = false) String error,
1417
            @RequestParam(value = "logout", required = false) String logout) {
1418
1419
1420
           ModelAndView model = new ModelAndView();
1421
           if (error != null) {
           model.addObject("error", "Invalid username and password!");
1422
1423
1424
1425
           if (logout != null) {
           model.addObject("msg", "You've been logged out successfully.");
1426
1427
1428
           model.setViewName("login");
1429
1430
           return model;
1431
1432
          }
1433
1434
         //for 403 access denied page
          @RequestMapping(value = "/403", method = RequestMethod.GET)
1435
1436
          public ModelAndView accesssDenied() {
1437
1438
           ModelAndView model = new ModelAndView();
1439
1440
           //check if user is login
1441
           Authentication auth = SecurityContextHolder.getContext().getAuthentication();
1442
           if (!(auth instanceof AnonymousAuthenticationToken)) {
1443
           UserDetails userDetail = (UserDetails) auth.getPrincipal();
           model.addObject("username", userDetail.getUsername());
1444
1445
1446
1447
           model.setViewName("403");
1448
           return model:
1449
1450
          }
          @RequestMapping(value = "/logout", method = RequestMethod.GET)
1451
1452
          public String logout(Model model) {
1453
           return "logout";
1454
1455
1456
        16)Test
1457
          -javaexpert/12345678은 admin과 일반 login모두 성공
1458
          -하지만 alex/12345678은 login 실패, 왜냐하면 role이 없기 때문에
          -login하려면 아래 query처럼 user_roles에 등록해야 ROLE_USER 권한을 부여 받음.
1459
           INSERT INTO user roles (user role id, username, role)
1460
1461
           VALUES (seg user role id.NEXTVAL, 'alex', 'ROLE USER');
1462
          -그렇다면 alex가 admin page에 접근할 수 있을까?
1463
          -현재는 403 error page로 이동한다. 왜냐하면 그는 ROLE_USER 권한만 있기 때문이다.
           HTTP Status 403 - Access is denied
1464
1465
           Username: alex
1466
           You do not have permission to access this page!
```