HOL : Spring Security

Task1. 간단한 인증처리하기

1. Spring Legacy Project 생성
   1)In Package Explorer > right-click > New > Other > Spring > Spring Legacy Project
   2)Project name : SpringSecurityDemo
   3)Select Spring MVC Project > Next
   4)com.example.biz > Finish

2. server.xml에 project 추가
   1)Tomcat v9.0 Server at localhost > right-click > Add and Remove
   2)SpringSecurityDemo > Add > Finish > OK

3. Spring security library download and install
   1)In pom.xml 아래와 같이 수정

```
      <java-version>1.8</java-version>
      <org.springframework-version>5.2.0.RELEASE</org.springframework-version>
      <org.aspectj-version>1.9.4</org.aspectj-version>
      <org.slf4j-version>1.7.28</org.slf4j-version>
      ...
      <!-- Servlet -->
      <dependency>
         <groupId>javax.servlet</groupId>
         <artifactId>javax.servlet-api</artifactId>
         <version>4.0.1</version>
         <scope>provided</scope>
      </dependency>
      <dependency>
         <groupId>javax.servlet.jsp</groupId>
         <artifactId>javax.servlet.jsp-api</artifactId>
         <version>2.3.3</version>
         <scope>provided</scope>
      </dependency>
      ...
      <!-- Test -->
      <dependency>
         <groupId>junit</groupId>
         <artifactId>junit</artifactId>
         <version>4.12</version>
         <scope>test</scope>
      </dependency>
```

   2)In mvnrepository, search 'spring security'
   3)Select 'Spring Security Core' 5.2.0
   4)Select 'Spring Security Web' 5.2.0
   5)Select 'Spring Security Config' 5.2.0
   6)Select 'Spring Security Taglibs' 5.2.0
   7)pom.xml > right-click > Run As > Maven clean and Maven install

4. DelegatingFilterProxy filter 구성
   1)In web.xml 에 code 추가

```
    <filter>
      <filter-name>springSecurityFilterChain</filter-name>
      <filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>
    </filter>

    <filter-mapping>
      <filter-name>springSecurityFilterChain</filter-name>
      <url-pattern>/*</url-pattern>
    </filter-mapping>
```

5. 보안 관련 설정 File 만들기

1)/src/main/webapp/WEB-INF/conig folder 생성
2)project > right-click > Build Path > Config Build Path
3)Source tab > Add Folder > select src/main/webapp/WEB-INF/
4)Click [Create New Folder] > Folder name : config > OK > Apply and Close
5)src/main/webapp/WEB-INF/config > right-click > New > Other > Springt > Spring Bean Configuration File
6)File name : securityContext.xml > Next
7)security check할 때 반드시 https://www.springframework.org/schema/security/spring-security.xsd 즉 xsd의 version이 없는 것을 선택해야 함.
8)Finish
9)In /WEB-INF/web.xml에 보안 관련 설정 file 등록

```
    <context-param>
      <param-name>contextConfigLocation</param-name>
      <param-value>classpath:securityContext.xml</param-value>
    </context-param>
```

6. Web 요청에 대한 보안 구성

1)In securityContext.xml code 추가

```
    <security:http auto-config="true">
      <security:intercept-url pattern="/login.html" access="hasRole('ROLE_USER')" />
      <security:intercept-url pattern="/welcome.html" access="hasRole('ROLE_ADMIN')" />
    </security:http>

    <security:authentication-manager>
      <security:authentication-provider>
        <security:user-service>
          <security:user name="javaexpert" password="{noop}12345678" authorities="ROLE_USER"/>
          <security:user name="admin" password="{noop}12345678" authorities="ROLE_ADMIN,ROLE_USER"/>
        </security:user-service>
      </security:authentication-provider>
    </security:authentication-manager>
```

7. web.xml 에서 코드 수정

```
    <servlet-mapping>
      <servlet-name>appServlet</servlet-name>
      <url-pattern>*.html</url-pattern>   <--여기 수정할 것
    </servlet-mapping>
```

8. Test

  1)/WEB-INF/views/security folder 생성

  2)/WEB-INF/views/security/login.jsp

```
      <body>
        <h1>login.jsp</h1>
      </body>
```

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

```
      <body>
        <h1>welcome.jsp</h1>
      </body>
```

  4)/WEB-INF/views/home.jsp

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

      </body>
      </html>
```

  5)HomeController.java 코드 수정

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

      @RequestMapping("/login.html")
      public String login(Model model) {
        return "security/login";
      }

      @RequestMapping("/welcome.html")
      public String welcome(Model model) {
        return "security/welcome";
      }
```

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

7)/login.html로 들어가면 정상적으로 login.jsp 로 이동한다.
8)username과 password를 미리 저장된 값을 넣지 않으면 Bad credentials로 로그인에러 발생한다.
9)/welcome.html로 들어가면 User와 Password를 입력할 수 있는 창으로 이동한다.
10)여기서 user와 password를 잘못 입력하면 'Bad credentials' 에러 메시지가 나온다.
11)User에 미리 설정한 javaexpert, Password에 12345678을 입력하면 403 Forbidden 오류가 나온다.
12)즉 권한이 없다는 뜻이다.
13)만일 미리 설정한 admin/12345678을 입력하면 정상적으로 welcome.jsp로 이동한다.
14)왜냐하면 이 계정은 ROLE_ADMIN 권한을 갖고 있기 때문이다.

Task2. Login page 만들기

1. securityContext.xml code 변경

```
<security:http auto-config="true">
   <security:form-login login-page="/loginForm.html"
      username-parameter="j_username"
      password-parameter="j_password"
      login-processing-url="/j_spring_security_check" />
   <security:intercept-url pattern="/login.html"
      access="hasRole('ROLE_USER')" />
   <security:intercept-url
      pattern="/welcome.html" access="hasRole('ROLE_ADMIN')" />
   <security:csrf disabled="true"/>
</security:http>
```

2. /views/security/loginForm.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
   pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>로그인 창</title>
</head>
<body>
   <h1>loginForm.jsp</h1>

   <form action="<c:url value="j_spring_security_check" />" method="post">
      ID : <input type="text" name="j_username"> <br />
      PW : <input type="text" name="j_password"> <br />
      <input type="submit" value="LOGIN"> <br />
   </form>
</body>
</html>
```

3. HomeController.java code 추가

```
@RequestMapping("/loginForm.html")
public String loginForm(Model model) {
   return "security/loginForm";
}
```

4. Test

1)project > right-click > Run As > Run on Server
2)http://localhost:8080/biz/login.html 을 요청하면 http://localhost:8080/biz/loginForm.html 로 redirect
3)ID와 Password를 미리 설정한 javaexpert/12345678을 넣으면 정상적으로 login.jsp로 이동한다.
4)http://localhost:8080/biz/welcome.html 을 요청하면 http://localhost:8080/biz/loginForm.html 로 redirect
5)만일 ID와 Password를 미리 설정한 javaexpert/12345678을 넣으면 403 forbidden error가 발생한다.
6)만일 ID와 Password를 미리 설정한 admin/12345678을 넣으면 정상적으로 welcome.jsp로 이동한다.
7)만일 잘못된 ID와 Password를 입력하면 계속 loginForm.html?error가 나타난다.

8)만일 Could not verify the provided CSRF token because your session was not found. 오류가 발생하면 아래의 조치를 취한다.

-securityContext.xml 에서 코드 추가한다.

```
<security:http auto-config="true">
   <security:form-login login-page="/loginForm.html" />
   <security:intercept-url pattern="/login.html"
      access="hasRole('ROLE_USER')" />
   <security:intercept-url
      pattern="/welcome.html" access="hasRole('ROLE_ADMIN')" />
   <security:csrf disabled="true"/>            <---cpde 추가
</security:http>
```

5. login 실패시 실패 message 출력하기

1)securityContext.xml code 추가

```
<security:form-login login-page="/loginForm.html"
   username-parameter="j_username"
   password-parameter="j_password"
   authentication-failure-url="/loginForm.html?ng"     <---code 추가
   login-processing-url="/j_spring_security_check" />
```

2)loginForm.jsp 코드 추가

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

```
        </c:if>
        ID : <input type="text" name="j_username"> <br />
        PW : <input type="text" name="j_password"> <br />
        <input type="submit" value="LOGIN"> <br />
    </form>
```

3)Test

-http://localhost:8080/biz/login.html 을 요청하면 http://localhost:8080/biz/loginForm.html 로 redirect

-ID와 Password를 미리 설정한 javaexpert/12345678을 넣으면 정상적으로 login.jsp로 이동한다.

-만일 잘못된 ID나 Password를 입력하면 아래와 같은 오류 message가 나온다.

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

-http://localhost:8080/biz/welcome.html 을 요청하면 http://localhost:8080/biz/loginForm.html 로 redirect

-만일 ID와 Password를 미리 설정한 javaexpert/12345678을 넣으면 403 forbidden error가 발생한다.

-만일 ID와 Password를 미리 설정한 admin/12345678을 넣으면 정상적으로 welcome.jsp로 이동한다.

-만일 잘못된 ID와 Password를 입력하면 계속 loginForm.html?ng가 나오면서 아래의 message가 나타난다.

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

Task3. login, logout 상태 표시하기

1. security/login.jsp page code 추가

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<body>
  <h1>login.jsp</h1>

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

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

2. Test

1)http://localhost:8080/biz/login.html 을 요청하면 http://localhost:8080/biz/loginForm.html 로 redirect

2)아이디와 패스워드를 미리 설정한 javaexpert/12345678을 넣으면 정상적으로 login.jsp로 이동한다.

3)이 때 화면에는 아래의 message가 나온다.

```
현재 로그인 상태
User ID :
org.springframework.security.authentication.UsernamePasswordAuthenticationToken@b09a9234:
Principal: org.springframework.security.core.userdetails.User@f486f6cc:
Username: javaexpert;
Password: [PROTECTED];
Enabled: true;
AccountNonExpired: true;
credentialsNonExpired: true;
AccountNonLocked: true;
Granted Authorities: ROLE_USER;
Credentials: [PROTECTED];
Authenticated: true;
Details: org.springframework.security.web.authentication.WebAuthenticationDetails@166c8:
RemoteIpAddress: 0:0:0:0:0:0:0:1;
SessionId: 949A2731FFC91E3C350818E0772C8E45;
Granted Authorities: ROLE_USER

Log Out
```

4)여기서 username만 출력하려면

User ID : ${pageContext.request.userPrincipal.name} 으로 변경한다.

3. securityContext.xml code 추가

```
  <security:form-login login-page="/loginForm.html"
    username-parameter="j_username"
    password-parameter="j_password"
    authentication-failure-url="/loginForm.html?ng"
    login-processing-url="/j_spring_security_check" />
  <security:logout logout-url="/j_spring_security_logout"/>     <--code 추가
```

4. Test

1)로그인 창에서 Log Out link를 클릭하면 index.html로 이동한다.

Task4. UserDetailsService 이용하기

1. Spring Legacy Project 생성

1)In Package Explorer > right-click > New > Other > Spring > Spring Legacy Project
2)Project name : SpringSecurityDemo1
3)Select Spring MVC Project > Next
4)com.example.biz > Finish

2. server.xml에 project 추가

1)Tomcat v9.0 Server at localhost > right-click > Add and Remove
2)SpringSecurityDemo1 > Add > Finish > OK

3. Spring security library download and install

1)In pom.xml 아래와 같이 수정

```
    <java-version>1.8</java-version>
    <org.springframework-version>5.2.0.RELEASE</org.springframework-version>
    <org.aspectj-version>1.9.4</org.aspectj-version>
    <org.slf4j-version>1.7.28</org.slf4j-version>
```

2)version 수정

```
    <dependency>
       <groupId>javax.servlet</groupId>
       <artifactId>javax.servlet-api</artifactId>
       <version>4.0.1</version>
       <scope>provided</scope>
    </dependency>
    <dependency>
       <groupId>javax.servlet.jsp</groupId>
       <artifactId>javax.servlet.jsp-api</artifactId>
       <version>2.3.3</version>
       <scope>provided</scope>
    </dependency>
    <dependency>
       <groupId>junit</groupId>
       <artifactId>junit</artifactId>
       <version>4.12</version>
       <scope>test</scope>
    </dependency>
```

3)In mvnrepository, search 'spring security'
   -Select 'Spring Security Core' 5.2.0.RELEASE
   -Select 'Spring Security Web' 5.2.0.RELEASE
   -Select 'Spring Security Config' 5.2.0.RELEASE
   -Select 'Spring Security Taglibs' 5.2.0.RELEASE

```
    <dependency>
       <groupId>org.springframework.security</groupId>
       <artifactId>spring-security-core</artifactId>
       <version>5.2.0.RELEASE</version>
    </dependency>
    <dependency>
       <groupId>org.springframework.security</groupId>
       <artifactId>spring-security-web</artifactId>
       <version>5.2.0.RELEASE</version>
    </dependency>
    <dependency>
       <groupId>org.springframework.security</groupId>
       <artifactId>spring-security-config</artifactId>
       <version>5.2.0.RELEASE</version>
    </dependency>
    <dependency>
       <groupId>org.springframework.security</groupId>
       <artifactId>spring-security-taglibs</artifactId>
       <version>5.2.0.RELEASE</version>
    </dependency>
```

4)pom.xml > right-click > Run As > Maven clean and Maven install

5)project > right-click > Properties > Project Facets > Java > Select 1.8

6)Runtimes Tab > Check Apache Tomcat v9.0

7)Click Apply and Close

4. HomeController.java 코드 수정

```
package com.example.biz;

import java.util.Date;

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

@Controller
public class HomeController {

   @RequestMapping(value = "/", method = RequestMethod.GET)
   public String home(Model model) {
      model.addAttribute("serverTime", new Date());
      return "home";
   }
}
```

5. project > right-click > Run As > Run on Server

6. src/main/webapp/WEB-INF/spring/Spring Bean Configuration File > spring-security.xml 파일 생성

1)check 'security'

2)security check할 때 반드시 https://www.springframework.org/schema/security/spring-security.xsd 즉 xsd의 version이 없는 것을 선택해야 함.

```
   <security:http auto-config="true" use-expressions="true">
   </security:http>

   <security:authentication-manager>
   </security:authentication-manager>
```

7. web.xml 코드 수정

```
<context-param>
   <param-name>contextConfigLocation</param-name>
   <param-value>/WEB-INF/spring/spring-security.xml</param-value>
</context-param>
...
...
<filter>
   <filter-name>springSecurityFilterChain</filter-name>
   <filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>
</filter>

<filter-mapping>
   <filter-name>springSecurityFilterChain</filter-name>
   <url-pattern>/*</url-pattern>
```

```
</filter-mapping>
```

8. WEB-INF/spring/appServlet/servlet-context.xml 수정

```
<context:component-scan base-package="com.example" />
```

9. project > right-click > Run As > Run on Server

10. spring-security.xml

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

  <security:http auto-config="true" use-expressions="true">
    <security:intercept-url pattern="/login"
      access="permitAll" />
    <security:intercept-url pattern="resources/**"
      access="permitAll" />
    <security:intercept-url pattern="/**"
      access="hasRole('ROLE_USER')" />
    <security:form-login login-page="/login"
      default-target-url="/welcome" username-parameter="j_username"
      password-parameter="j_password"
      authentication-failure-url="/login?error"
      login-processing-url="/j_spring_security_check"
      always-use-default-target="true" />
    <security:logout invalidate-session="true"
      logout-url="/logout" logout-success-url="/login?logout" />
    <security:csrf />
  </security:http>

  <security:authentication-manager>
    <security:authentication-provider user-service-ref="memberService" />
  </security:authentication-manager>

  <bean id="memberService"
    class="com.example.service.MemberService">
    <property name="userService" ref="userService" />
  </bean>
  <bean id="userService"
    class="com.example.service.UserService">
    <property name="userDao" ref="userDao" />
  </bean>
  <bean id="userDao" class="com.example.dao.UserDao">
    <property name="username" value="{noop}javaexpert" />
```

```
        <property name="password" value="{noop}12345678" />
    </bean>
</beans>
```

11. HomeController.java

```java
package com.example.biz;

import java.util.Date;

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

@Controller
public class HomeController {

    @RequestMapping(value = "/", method = RequestMethod.GET)
    public String home(Model model) {
        model.addAttribute("serverTime", new Date());
        return "home";
    }

    @RequestMapping(value = "/login", method = RequestMethod.GET)
    public ModelAndView login(
        @RequestParam(value = "error", required = false) String error,
        @RequestParam(value = "logout", required = false) String logout) {
        ModelAndView model = new ModelAndView();
        if (error != null) {
            model.addObject("error", "Invalid username and password!");
        }
        if (logout != null) {
            model.addObject("msg", "You've been logged out successfully.");
        }
        model.setViewName("login");
        return model;
    }

    @RequestMapping("/welcome")
    public String welcome(Model model) {
        return "welcome";
    }

    @RequestMapping(value = "/logout", method = RequestMethod.GET)
    public String logout(Model model) {
        return "logout";
    }
}
```

12. com.example.vo.UserVO.java

```
package com.example.vo;

public class UserVO {
    private String username;
    private String password;

    public UserVO() {}

    public UserVO(String username, String password) {
        this.username = username;
        this.password = password;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    @Override
    public String toString() {
        return "UserVO [username=" + username + ", password=" + password + "]";
    }
}
```

13. com.example.dao.UserDao.java

```
package com.example.dao;

import org.springframework.stereotype.Repository;

import com.example.vo.UserVO;

@Repository("userDao")
public class UserDao {
    private String username;
    private String password;

    public void setUsername(String username) {
        this.username = username;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    public UserVO getUsersByID(String username) {
        return new UserVO(this.username, this.password);
    }
}
```

14. com.example.service.UserService.java

```
package com.example.service;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.example.dao.UserDao;
import com.example.vo.UserVO;

@Service("userService")
public class UserService {
    private UserDao userDao;

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

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

15. com.example.service.MemberService.java

```
package com.example.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.example.vo.UserVO;

@Service("memberService")
public class MemberService implements UserDetailsService{
  private UserService userService;

  /*
   * 사용자가 입력한 ID/PWD 를 검증하기위해 저장소에 그 ID/PWD 가 있는지 확인하는 source이다. 저장소는 in-memory가 될 수도있고,
   * DB가 될 수도있고 어떠한 것(Facebook,Naver등등)도 될 수있다. userService는 DB 에 접근하기 위한 service이다.
   */

  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);
    return user;
  }
}
```

16. views/login.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
  pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
<html>
<head>
  <title>로그인 페이지</title>
  <meta charset="UTF-8">
  <script>
    function doLogin() {
      if(frm.j_username.value == "") {
        alert("아이디를 입력해주세요.");
        return;
      }
      if(frm.j_password.value == "") {
        alert("패스워드를 입력해주세요.");
        return;
      }
      frm.submit();
    }
  </script>
</head>
<body>
  <form name="frm" action="j_spring_security_check" method="post">
    <ul>
      <li>
        <label for="userID">ID</label>
        <input id = "userID" type="userID" name="j_username" placeholder="ID" required>
      </li>
      <li>
        <label for="password">Password</label>
        <input id = "password" type="password" name="j_password" placeholder="PASSWORD" required></li>
      <li>
        <input type="button" value="로그인" onclick="doLogin()"/>
      </li>
    </ul>
    <input type="hidden" name="${_csrf.parameterName}"
      value="${_csrf.token}" />
  </form>

  <c:if test="${not empty error}">
    <div class="error">${error}</div>
  </c:if>
  <c:if test="${not empty msg}">
    <div class="msg">${msg}</div>
  </c:if>
</body>
</html>
```

17. views/welcome.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
  pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Member Page</title>
</head>
<body>
  <c:if test="${pageContext.request.userPrincipal.name != null}">
    <h2>
      Welcome : ${pageContext.request.userPrincipal.name} |
      <a href="logout"> Logout</a>
    </h2>
  </c:if>
```

```
</body>
</html>
```

18. views/logout.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
  pageEncoding="UTF-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<c:set var="username" value="${pageContext.request.userPrincipal.name}" />
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Logout page</title>
</head>
<body>
  <c:url value="/logout" var="logoutUrl" />
  <!-- csrt for log out-->
  <form action="${logoutUrl}" method="post" id="logoutForm">
    <input type="hidden" name="${_csrf.parameterName}"
     value="${_csrf.token}" />
  </form>
  <script>
    function formSubmit() {
      document.getElementById("logoutForm").submit();
    }
  </script>
  <form>
    <span style="color: gray;" ><h5> ${username} 님 반갑습니다.
    <a href = "javascript:formSubmit()"> 로그아웃 </a> </h5></span>
  </form>
</body>
</html>
```

19. Test
  1)SpringSecurityDemo1 > right-click > Run As > Run on Server
  2)http://localhost:8080/biz/login 으로 이동되며
  3)이때 ID와 Password를 잘 못 입력하면 Invalid username and password! 를 보며
  4)javaexpert/12345678을 입력하면 welcome으로 이동한다.
  5)welcome에서 Logout link click하면
  6)logout으로 이동하고 [로그아웃] link를 click하면 다시 첫 page로 이동하지만, You've been logged out successfully. message를 만난다.

Task5. 보안관련 taglibs 사용하기
1. SpringSecurityDemo project에서
2. pom.xml 에서 spring-security-taglibs : 5.2.0.RELEASE 확인
3. security/login.jsp에서
  1)아래 taglib 추가

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

2)코드 변경

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

3)위의 코드를 아래의 코드로 변경

```
    <security:authorize access="hasRole('ROLE_USER')">
      <p>현재 로그인 상태</p>
    </security:authorize>
    <security:authorize access="hasRole('ROLE_ANONYMOUS')">
      <p>현재 로그아웃 상태</p>
    </security:authorize>
```

4)아래의 코드도 변경

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

5)아래의 코드로 변경

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

4. Test

1)http://localhost:8080/biz/login.html 을 요청하면 http://localhost:8080/biz/loginForm.html 로 redirect
2)아이디와 패스워드를 미리 설정한 javaexpert/12345678을 넣으면 정상적으로 login.jsp로 이동한다.
3)이 때 화면에는 아래의 message가 나온다.

```
    login.jsp
    현재 로그인 상태
    User ID : javaexpert
    Log Out
```

4)Logout link를 click하면 다시 첫 page로 이동한다.

Task6. Database-based Login Authentication

1. Spring Legacy Project 생성

1)In Package Explorer > right-click > New > Other > Spring > Spring Legacy Project
2)Project name : SpringSecurityDatabaseDemo
3)Select Spring MVC Project > Next
4)com.example.biz > Finish

2. server.xml에 project 추가

1)Tomcat v9.0 Server at localhost > right-click > Add and Remove
2)SpringSecurityDatabaseDemo > Add > Finish > OK

3. Spring security library download and install

1)In pom.xml 아래와 같이 수정

```
<java-version>1.8</java-version>
<org.springframework-version>5.2.0.RELEASE</org.springframework-version>
<org.aspectj-version>1.9.4</org.aspectj-version>
<org.slf4j-version>1.7.28</org.slf4j-version>
```

2)version 수정

```
<dependency>
   <groupId>javax.servlet</groupId>
   <artifactId>javax.servlet-api</artifactId>
   <version>4.0.1</version>
   <scope>provided</scope>
</dependency>
<dependency>
   <groupId>javax.servlet.jsp</groupId>
   <artifactId>javax.servlet.jsp-api</artifactId>
   <version>2.3.3</version>
   <scope>provided</scope>
</dependency>
<dependency>
   <groupId>junit</groupId>
   <artifactId>junit</artifactId>
   <version>4.12</version>
   <scope>test</scope>
</dependency>
```

3)In mvnrepository, search 'spring security'

-Select 'Spring Security Core' 5.2.0.RELEASE
-Select 'Spring Security Web' 5.2.0.RELEASE
-Select 'Spring Security Config' 5.2.0.RELEASE
-Select 'Spring Security Taglibs' 5.2.0.RELEASE

```
<dependency>
   <groupId>org.springframework.security</groupId>
   <artifactId>spring-security-core</artifactId>
   <version>5.2.0.RELEASE</version>
</dependency>
<dependency>
   <groupId>org.springframework.security</groupId>
   <artifactId>spring-security-web</artifactId>
   <version>5.2.0.RELEASE</version>
</dependency>
<dependency>
   <groupId>org.springframework.security</groupId>
   <artifactId>spring-security-config</artifactId>
   <version>5.2.0.RELEASE</version>
</dependency>
<dependency>
   <groupId>org.springframework.security</groupId>
   <artifactId>spring-security-taglibs</artifactId>
```

```
      <version>5.2.0.RELEASE</version>
    </dependency>
```

4)pom.xml > right-click > Run As > Maven clean and Maven install
5)project > right-click > Properties > Project Facets > Java > Select 1.8
6)Runtimes Tab > Check Apache Tomcat v9.0
7)Click Apply and Close

4. HomeController.java

```
package com.example.biz;

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

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

5. home.jsp

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

</body>
</html>
```

6. project > right-click > Run As > Run on Server

7. Database table & sequence creation

```
CREATE SEQUENCE seq_user_role_id
MAXVALUE 9999;
/

CREATE  TABLE Users (
    username VARCHAR2(20),
    password VARCHAR2(20) NOT NULL,
    enabled NUMBER(1) DEFAULT 1 ,
    CONSTRAINT users_username_pk PRIMARY KEY (username)
  );

  CREATE TABLE User_roles (
    user_role_id  NUMBER(4),
    username VARCHAR2(45),
    role VARCHAR2(45) NOT NULL,
    CONSTRAINT user_roles_id PRIMARY KEY (user_role_id),
    CONSTRAINT user_roles_uk UNIQUE (role,username),
    CONSTRAINT user_roles_fk FOREIGN KEY (username) REFERENCES users (username)
  );
```

8. Inserts some records for testing.

```
  INSERT INTO Users(username,password,enabled) VALUES ('javaexpert','12345678', 1);
  INSERT INTO users(username,password,enabled) VALUES ('alex','12345678', 1);

  INSERT INTO user_roles (user_role_id, username, role)
  VALUES (seq_user_role_id.NEXTVAL, 'javaexpert', 'ROLE_USER');

  INSERT INTO user_roles (user_role_id, username, role)
  VALUES (seq_user_role_id.NEXTVAL, 'javaexpert', 'ROLE_ADMIN');

  COMMIT;
```

9. pom.xml에 oracle driver와 MyBatis 관련 dependency 추가

```
  <dependency>
    <groupId>com.oracle</groupId>
    <artifactId>ojdbc6</artifactId>
    <version>11.2</version>
  </dependency>
  <!-- https://mvnrepository.com/artifact/org.mybatis/mybatis -->
  <dependency>
    <groupId>org.mybatis</groupId>
    <artifactId>mybatis</artifactId>
    <version>3.5.3</version>
  </dependency>
  <!-- https://mvnrepository.com/artifact/org.mybatis/mybatis-spring -->
  <dependency>
    <groupId>org.mybatis</groupId>
    <artifactId>mybatis-spring</artifactId>
    <version>2.0.3</version>
  </dependency>
```

10. Database 연결을 위한 환경설정

1)Build path에 config foler 추가

-project right-click > Build Path > Configure Build Path > Select [Source] tab

-Click [Add Folder] > Select 현재 project > Click [Create New Folder...]

-Folder name : config > Finish > OK > Apply and Close

-Java Resources > config folder 확인

2)config/dbinfo.properties file 생성

```
db.driver=oracle.jdbc.driver.OracleDriver
db.url=jdbc:oracle:thin:@localhost:1521:XE
db.username=hr
db.password=hr
```

3)pom.xml에 spring-jdbc 추가

```
<!-- https://mvnrepository.com/artifact/org.springframework/spring-jdbc -->
<dependency>
   <groupId>org.springframework</groupId>
   <artifactId>spring-jdbc</artifactId>
   <version>5.2.0.RELEASE</version>
</dependency>
```

11. web.xml 코드 추가

```
<filter>
   <filter-name>springSecurityFilterChain</filter-name>
   <filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>
</filter>

<filter-mapping>
   <filter-name>springSecurityFilterChain</filter-name>
   <url-pattern>/*</url-pattern>
</filter-mapping>
```

12. web.xml 수정

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

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

1)src/main/webapp/WEB-INF/spring > right-click > New > Spring Bean Configuration File

2)File name : spring-security.xml > Next

3)Check beans, security, context > Finish

4)security check할 때 반드시 https://www.springframework.org/schema/security/spring-security.xsd 즉 xsd의 version이 없는 것을 선택해야 함.

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

```
  xmlns:context="http://www.springframework.org/schema/context"
  xsi:schemaLocation="http://www.springframework.org/schema/security http://www.springframework.org/schema/security/spring-security.xsd
    http://www.springframework.org/schema/beans http://www.springframework.org/schema/beans/spring-beans.xsd">

  <context:property-placeholder location="classpath:dbinfo.properties"/>
  <bean id="dataSource"
    class="org.springframework.jdbc.datasource.DriverManagerDataSource">
    <property name="driverClassName"
      value="${db.driver}" />
    <property name="url"
      value="${db.url}" />
    <property name="username" value="${db.username}" />
    <property name="password" value="${db.password}" />
  </bean>
  <bean id="customPasswordEncoder"
    class="com.example.security.MyPasswordEncoder" />
  <!-- enable use-expressions -->
  <security:http>

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

    <!-- access denied page -->
    <security:access-denied-handler error-page="/403" />

    <security:form-login
      login-page="/login"
      default-target-url="/welcome"
      authentication-failure-url="/login?error"
      login-processing-url="/j_spring_security_check"
      username-parameter="username"
      password-parameter="password" />
    <security:logout invalidate-session="true"
      logout-url="/j_spring_security_logout"   logout-success-url="/login?logout" />
    <!-- enable csrf protection -->
    <security:csrf   disabled="true"/>
  </security:http>

  <!-- Select users and user_roles from database -->
  <security:authentication-manager>
   <security:authentication-provider>
    <security:jdbc-user-service data-source-ref="dataSource"
     users-by-username-query=
      "select username,password, enabled from users where username=?"
     authorities-by-username-query=
      "select username, role from user_roles where username =?  " />
    <security:password-encoder ref="customPasswordEncoder" />
   </security:authentication-provider>
  </security:authentication-manager>
</beans>

```

14. views/hello.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
  pageEncoding="UTF-8"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<%@taglib prefix="sec"
  uri="http://www.springframework.org/security/tags"%>
<html>
<body>
  <h1>Title : ${title}</h1>
  <h1>Message : ${message}</h1>

  <sec:authorize access="hasRole('ROLE_USER')">
    <!-- For login user -->
    <c:url value="/j_spring_security_logout" var="logoutUrl" />
    <form action="${logoutUrl}" method="post" id="logoutForm">
      <input type="hidden" name="${_csrf.parameterName}"
        value="${_csrf.token}" />
    </form>
    <script>
      function formSubmit() {
        document.getElementById("logoutForm").submit();
      }
    </script>

    <c:if test="${pageContext.request.userPrincipal.name != null}">
      <h2>
        User : <sec:authentication property="name" /> | <a
          href="javascript:formSubmit()"> Logout</a>
      </h2>
    </c:if>


  </sec:authorize>
</body>
</html>
```

15. views/login.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
  pageEncoding="UTF-8"%>
  <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<html>
<head>
<title>Login Page</title>
<style>
  .error {
    padding: 15px;
    margin-bottom: 20px;
    border: 1px solid transparent;
```

```
      border-radius: 4px;
      color: #a94442;
      background-color: #f2dede;
      border-color: #ebccd1;
    }

    .msg {
      padding: 15px;
      margin-bottom: 20px;
      border: 1px solid transparent;
      border-radius: 4px;
      color: #31708f;
      background-color: #d9edf7;
      border-color: #bce8f1;
    }

    #login-box {
      width: 300px;
      padding: 20px;
      margin: 100px auto;
      background: #fff;
      -webkit-border-radius: 2px;
      -moz-border-radius: 2px;
      border: 1px solid #000;
    }
  </style>
  </head>
  <body onload='document.loginForm.username.focus();'>

    <h1>Spring Security Login Form (Database Authentication)</h1>

    <div id="login-box">

      <h2>Login with Username and Password</h2>

      <c:if test="${not empty error}">
        <div class="error">${error}</div>
      </c:if>
      <c:if test="${not empty msg}">
        <div class="msg">${msg}</div>
      </c:if>

      <form name='loginForm'
        action="<c:url value='/j_spring_security_check' />" method='POST'>

      <table>
        <tr>
          <td>User:</td>
          <td><input type='text' name='username'></td>
        </tr>
        <tr>
          <td>Password:</td>
          <td><input type='password' name='password' /></td>
        </tr>
        <tr>
          <td colspan='2'><input name="submit" type="submit"
           value="submit" /></td>
        </tr>
        </table>

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

      </form>
    </div>
  </body>
  </html>
```

16. views/admin.jsp

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

  <c:url value="/j_spring_security_logout" var="logoutUrl" />
  <form action="${logoutUrl}" method="post" id="logoutForm">
    <input type="hidden" name="${_csrf.parameterName}"
      value="${_csrf.token}" />
  </form>
  <script>
    function formSubmit() {
      document.getElementById("logoutForm").submit();
    }
  </script>

  <c:if test="${pageContext.request.userPrincipal.name != null}">
    <h2>
      Welcome : ${pageContext.request.userPrincipal.name} | <a
        href="javascript:formSubmit()"> Logout</a>
    </h2>
  </c:if>

</body>
</html>
```

17. views/403.jsp

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

```
   pageEncoding="UTF-8"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<html>
<body>
   <h1>HTTP Status 403 - Access is denied</h1>

   <c:choose>
      <c:when test="${empty username}">
        <h2>You do not have permission to access this page!</h2>
      </c:when>
      <c:otherwise>
        <h2>Username : ${username} <br/>
              You do not have permission to access this page!</h2>
      </c:otherwise>
   </c:choose>

</body>
</html>
```

18. HomeController.java

```
@RequestMapping(value = { "/", "/welcome**" }, method = RequestMethod.GET)
public ModelAndView defaultPage() {

  ModelAndView model = new ModelAndView();
  model.addObject("title", "Spring Security Login Form - Database Authentication");
  model.addObject("message", "This is default page!");
  model.setViewName("hello");
  return model;

}

@RequestMapping(value = "/admin**", method = RequestMethod.GET)
public ModelAndView adminPage() {

  ModelAndView model = new ModelAndView();
  model.addObject("title", "Spring Security Login Form - Database Authentication");
  model.addObject("message", "This page is for ROLE_ADMIN only!");
  model.setViewName("admin");
  return model;

}

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

  ModelAndView model = new ModelAndView();
  if (error != null) {
   model.addObject("error", "Invalid username and password!");
  }
  if (logout != null) {
   model.addObject("msg", "You've been logged out successfully.");
  }
  model.setViewName("login");

  return model;

}

//for 403 access denied page
@RequestMapping(value = "/403", method = RequestMethod.GET)
public ModelAndView accesssDenied() {

  ModelAndView model = new ModelAndView();

  //check if user is login
  Authentication auth = SecurityContextHolder.getContext().getAuthentication();
  if (!(auth instanceof AnonymousAuthenticationToken)) {
   UserDetails userDetail = (UserDetails) auth.getPrincipal();
   model.addObject("username", userDetail.getUsername());
  }

  model.setViewName("403");
  return model;

}
@RequestMapping(value = "/logout", method = RequestMethod.GET)
public String logout(Model model) {
   return "logout";
}
```

19. com.example.security.MyPasswordEncoder Class 생성

  1)com.example.security package 생성

  2)com.example.security.MyPasswordEncoder Class

```
package com.example.security;

import org.springframework.security.crypto.password.PasswordEncoder;

public class MyPasswordEncoder implements PasswordEncoder {

   @Override
   public String encode(CharSequence rawPassword) {
      return rawPassword.toString();
   }

   @Override
   public boolean matches(CharSequence rawPassword, String encodedPassword) {
      return rawPassword.toString().equals(encodedPassword);
   }
```

```
    }
```

20. Test

1)javaexpert/12345678은 admin과 일반 login모두 성공

2)하지만 alex/12345678은 login 실패, 왜냐하면 role이 없기 때문에

3)login하려면 아래 query처럼 user_roles에 등록해야 ROLE_USER 권한을 부여 받음.

```
INSERT INTO user_roles (user_role_id, username, role)
VALUES (seq_user_role_id.NEXTVAL, 'alex', 'ROLE_USER');
```

4)그렇다면 alex가 admin page에 접근할 수 있을까?

5)현재는 403 error page로 이동한다. 왜냐하면 그는 ROLE_USER 권한만 있기 때문이다.

```
HTTP Status 403 - Access is denied
Username : alex
You do not have permission to access this page!
```