Spring Security

1. 소개

1)웹 보안 개요

-All-in-One Java 애플리케이션 개발, 전병선, 2014 참조

-웹 어플리케이션을 작성하면서 보안과 관련해서 고려할 사항은 다음과 같다.

--인증(authentication) : 사용자가 자신임을 증명하는 속성
--권한(authorization) : 사용자가 어떤 리소스에 접근하게 할지를 결정하는 속성
--기밀성(privacy or confidentiality) : 권한이 없는 사용자에게 접근을 허용하지 않게 하는 속성
--무결성(integrity) : 권한이 없는 사용자가 데이터를 변경하기 못하게 하는 속성
--부인방지(repudiation) : 사용자가 자신이 했다는 것을 부인하지 못하게 하는 속성

2)Spring Security는 이들 보안 요소 중에서 인증과 권한 서비스를 제공한다.
3)인증은 사용자가 자신임을 증명하는 것이고, 권한은 사용자가 어떤 리소스에 접근할 수 있는지를 결정하는 것이다.
4)우리가 웹 어플리케이션의 리소스에 접근할 수 있는지 여부는 권한과 관련되어 있다.
5)사용자가 권한이 있는지를 알기 위해서는 권한을 가진 사용자인지 인증되어야 한다.
6)그러니까 먼저 사용자가 인증되어야 하고, 인증된 사용자에 부여된 권한으로 리소스에 접근할 수 있는지를 체크한다.
7)사용자를 인증하는 다양한 방법이 있지만 가장 보편적으로 사용하는 방법은 폼 인증 방식이다.
8)사용자가 인증된 후에는 리소스에 접근할 수 있는 권한을 가졌는지를 체크해야 한다.
9)사용자의 권한을 수립하는 첫 번째 단계는 principle 형식으로 사용자를 표현하는 것이다.
10)principle은 리소스에 접근하기 위한 보안 식별자가 할당된 계정 보유자로서, 사용자, 그룹, 서비스, 컴퓨터 등이 principle이 될 수 있다.
11)개별 사용자에 대해 일일이 권한을 가졌는지를 검사하는 것을 비효율적일 뿐만 아니라 경우에 따라서는 불가능하기도 하다.
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.

2. 환경설정

1)Spring Legacy Project 생성

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

2)server.xml에 project 추가

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

3)Spring security library download and install

-In pom.xml 아래와 같이 수정

```
<java-version>1.8</java-version>
<org.springframework-version>4.3.17.RELEASE</org.springframework-version>
<org.aspectj-version>1.9.1</org.aspectj-version>
<org.slf4j-version>1.6.6</org.slf4j-version>
```

-In mvnrepository, search 'spring security'
-Select 'Spring Security Core' 4.2.6
-Select 'Spring Security Web' 4.2.6
-Select 'Spring Security Config' 4.2.6
-Select 'Spring Security Taglibs' 4.2.6
-pom.xml > right-click > Run As > Maven clean and Maven install

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

5)보안 관련 설정 파일 만들기

-/src/main/webapp/WEB-INF/conig folder 생성
-project > right-click > Build Path > Config Build Path

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

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

-In securityContext.xml 코드 추가

```
<security:http auto-config="true">
    <!--auto-config='true'를 설정한 것만으로 기본 로그인페이지/HTTP 기본인증/로그아웃
    기능등을 제공한다. -->
    <!--만일 여기서 use-expressions='true'라고 하면 SpEL을 사용한다는 의미이다. -->
    <!-- use-expressions은 기본값이 false이다. 이럴때에는 SpEL을 사용하지 않는다는
    의미이다. -->
    <security:intercept-url pattern="/login.html"
    access="hasRole('ROLE_USER')" />
    <security:intercept-url
    pattern="/welcome.html" access="hasRole('ROLE_ADMIN')" />
</security:http>
```

```
--<intercept-url pattern="..." access="ROLE_ANONYMOUS" />
--<intercept-url pattern="..." access="IS_AUTHENTICATED_ANONYMOUSLY" />
--<intercept-url pattern="..." access="ROLE_USER" />
--<intercept-url pattern="..." access="ROLE_ADMIN" />
--해당 URL 에 접근하기위한 권한을 설정하여준다.
--접근가능한 IP 등을 설정할 수도 있다.
--그리고 권한은 위쪽이 우선시된다.
--<intercept-url pattern="/login" access="permitAll" />
    ---/login 으로는 모두 허용해준다./login 을 막아놓으면 안되니까.
--<intercept-url pattern="/resources/**" access="permitAll" />
    ---리소스도 허용
--<intercept-url pattern="/**" access="hasRole('ROLE_USER')" />
    ---나머지는 모두 ROLE_USER 권한을 가진사람만 허용해준다.
```

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

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

-In web.xml 에 코드 추가
    --반드시 filter의 이름은 springSecurityFilterChain 이어야 한다.

```
<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>
```

-In web.xml 에서 코드 수정

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

6) Test

-/WEB-INF/views/security folder 생성

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

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

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

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

-/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>
```

-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";
        }
```

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

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

3. Login page 만들기

1)securityContext.xml 파일 코드 추가

```
    <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>
```

-form 기반 인증을 사용하고자 할 때는 아래와 같이 설정한다.

```
        <form-login login-page="/login"                -----사용자가 만든 로그인페이지를 스프링에게
        알려준다.  이거 설정안하면 스프링이 만들어준것을 사용.
            default-target-url="/monitering"           -----로그인성공하면 이동할 페이지설정
            username-parameter="username"
            password-parameter="password"
            authentication-failure-url="/login?error"  -----실패시 호출해줄 URL (login 페이지에
            error 파라미터를 보내준다)
            always-use-default-target='true'            이거 안하면 로그인 성공해도 /monitering 로
            제대로 안갈 수 있다.
        />
        <logout invalidate-session="true" logout-url="/logout"
        logout-success-url="/login?logout" />
            --로그아웃되면 세션을 초기화한다.
            --logout-success-url="/login?logout"      ------- 로그아웃되면 이동할 페이지
            --logout-url="/logout"                    --------로그아웃을 위한 URL설정. 이거
            안해주면 디폴트로 j_spring_security_logout 해주면됨.
        <csrf/>   ---------- 간단한 설정으로 csrf 를 통한 해킹을 막을수있다.
         ( CSRF 설명:
         http://tm-csc.tistory.com/entry/7-WebHacking%EC%9B%B9%ED%95%B4%ED%82
         %B9-CSRF)
```

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 코드 추가

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

4)Test

- -project > right-click > Run As > Run on Server
- -http://localhost:8080/biz/login.html 을 요청하면 http://localhost:8080/biz/loginForm.html 로 redirect
- -아이디와 패스워드를 미리 설정한 javaexpert/12345678을 넣으면 정상적으로 login.jsp로 이동한다.
- -http://localhost:8080/biz/welcome.html 을 요청하면 http://localhost:8080/biz/loginForm.html 로 redirect
- -만일 아이디와 패스워드를 미리 설정한 javaexpert/12345678을 넣으면 403 forbidden 에러가 발생한다.
- -만일 아이디와 패스워드를 미리 설정한 admin/12345678을 넣으면 정상적으로 welcome.jsp로 이동한다.
- -만일 잘못된 아이디와 패스워드를 입력하면 계속 loginForm.html?error가 나타난다.

-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"/>           <---코드 추가
			</security:http>
```

4. login 실패시 실패 메시지 출력하기

1)securityContext.xml 코드 추가

```
	<security:form-login login-page="/loginForm.html"
		username-parameter="j_username"
		password-parameter="j_password"
		authentication-failure-url="/loginForm.html?ng"    <---코드 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
-아이디와 패스워드를 미리 설정한 javaexpert/12345678을 넣으면 정상적으로 login.jsp로 이동한다.
-만일 잘못된 아이디나 패스워드를 입력하면 아래와 같은 오류 메시지가 나온다.

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

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

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

5. login, logout 상태 표시하기

1)login.jsp 페이지 코드 추가

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

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

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

2)Test

-http://localhost:8080/biz/login.html 을 요청하면 http://localhost:8080/biz/loginForm.html 로 redirect
-아이디와 패스워드를 미리 설정한 javaexpert/12345678을 넣으면 정상적으로 login.jsp로 이동한다.
-이 때 화면에는 아래의 메시지가 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
```

3)여기서 username만 출력하려면

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

4)securityContext.xml 코드 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"/>    <--코드 추가
```

5)Test
-로그인 창에서 Log Out link를 클릭하면 index.html로 이동한다.

6. Lab
1)Spring Legacy Project 생성
-In Package Explorer > right-click > New > Spring Legacy Project
-Project name : SpringSecurityDemo
-Select Spring MVC Project > Next
-com.javasoft.biz > Finish

2)server.xml에 project 추가
-Tomcat v8.5 Server at localhost > right-click > Add and Remove
-SpringSecurityDemo > Add > Finish > OK

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

```
<java-version>1.8</java-version>
<org.springframework-version>4.3.17.RELEASE</org.springframework-version>
```

-In mvnrepository, search 'spring security'
-Select 'Spring Security Core' 4.2.6
-Select 'Spring Security Web' 4.2.6
-Select 'Spring Security Config' 4.2.6
-Select 'Spring Security Taglibs' 4.2.6
-pom.xml > right-click > Run As > Maven clean and Maven install

4)HomeController.java 코드 수정

```
package com.javasoft.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-security.xml 파일 생성
-check 'security'

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)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-4.2.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>
                <security:user-service>
                    <security:user name="javaexpert" password="12345678"
                        authorities="ROLE_USER" />
                    <security:user name="admin" password="12345678"
                        authorities="ROLE_ADMIN,ROLE_USER" />
                </security:user-service>
            </security:authentication-provider> -->
            <security:authentication-provider user-service-ref="memberService" />
        </security:authentication-manager>

        <bean id="memberService"
            class="com.javasoft.service.MemberService">
            <property name="userService" ref="userService" />
        </bean>
        <bean id="userService"
            class="com.javasoft.service.UserService">
            <property name="userDao" ref="userDao" />
        </bean>
        <bean id="userDao" class="com.javasoft.dao.UserDao">
            <property name="username" value="javaexpert" />
            <property name="password" value="12345678" />
        </bean>
    </beans>
```

9)HomeController.java

```
    package com.javasoft.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";
        }
    }
```

10)com.javasoft.vo.UserVO.java

```
    package com.javasoft.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 + "]";
        }
    }
```

11)com.javasoft.dao.UserDao.java

```
    package com.javasoft.dao;

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

    import com.javasoft.vo.UserVO;

    @Repository("userDao")
    public class UserDao {
        @Autowired
        private UserVO userVO;
        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);
        }
    }
```

12)com.javasoft.service.UserService.java

```
    package com.javasoft.service;

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

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

    public class UserService {
        private UserDao userDao;

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

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

13)com.javasoft.service.MemberService.java

```
package com.javasoft.service;

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

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

import com.javasoft.vo.UserVO;

public class MemberService implements UserDetailsService{
    private UserService userService;


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

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

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

14)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>
```

15)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>
```

16)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>
```

7. 보안관련 taglibs 사용하기
   1)pom.xml 에서 spring-security-taglibs : 4.2.6.RELEASE 확인
   2)login.jsp에서
      -아래 taglib 추가

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

      -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>
```

      -위의 코드를 아래의 코드로 변경

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

-코드 변경

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

-아래의 코드로 변경

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

8. Database-based Login Authentication

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

2)server.xml에 project 추가
- -Tomcat v8.5 Server at localhost > right-click > Add and Remove
- -SpringSecurityDatabaseDemo > Add > Finish > OK

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

```
<java-version>1.8</java-version>
<org.springframework-version>4.3.17.RELEASE</org.springframework-version>
```

- -In mvnrepository, search 'spring security'
- -Select 'Spring Security Core' 4.2.6
- -Select 'Spring Security Web' 4.2.6
- -Select 'Spring Security Config' 4.2.6
- -Select 'Spring Security Taglibs' 4.2.6
- -pom.xml > right-click > Run As > Maven clean and Maven install

4)HomeController.java

```
package com.javasoft.biz;

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

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

5)home.jsp

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

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

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

7)Database table 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');
```

9)pom.xml에 oracle driver와 MyBatis 관련 dependency 추가

```
    <dependency>
        <groupId>com.oracle</groupId>
        <artifactId>ojdbc8</artifactId>
        <version>12.2</version>
    </dependency>
    <!-- https://mvnrepository.com/artifact/org.mybatis/mybatis -->
    <dependency>
        <groupId>org.mybatis</groupId>
        <artifactId>mybatis</artifactId>
        <version>3.4.6</version>
    </dependency>
    <!-- https://mvnrepository.com/artifact/org.mybatis/mybatis-spring -->
    <dependency>
        <groupId>org.mybatis</groupId>
        <artifactId>mybatis-spring</artifactId>
        <version>1.3.2</version>
    </dependency>
```

10)Database 연결을 위한 환경설정

-pom.xml에 spring-jdbc 추가

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

-web.xml 코드 추가

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

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

-web.xml 수정

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

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

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

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

--Check beans, security > Finish

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

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

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

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

        <!-- 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:authentication-provider>
    </security:authentication-manager>
</beans>
```

11)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 : ${pageContext.request.userPrincipal.name} | <a
                    href="javascript:formSubmit()"> Logout</a>
            </h2>
        </c:if>
    </sec:authorize>
</body>
</html>

12)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>
```

13)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>
```

14)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>
```

15)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";
        }
```