Spring security 따라하기

한성곤 커미터

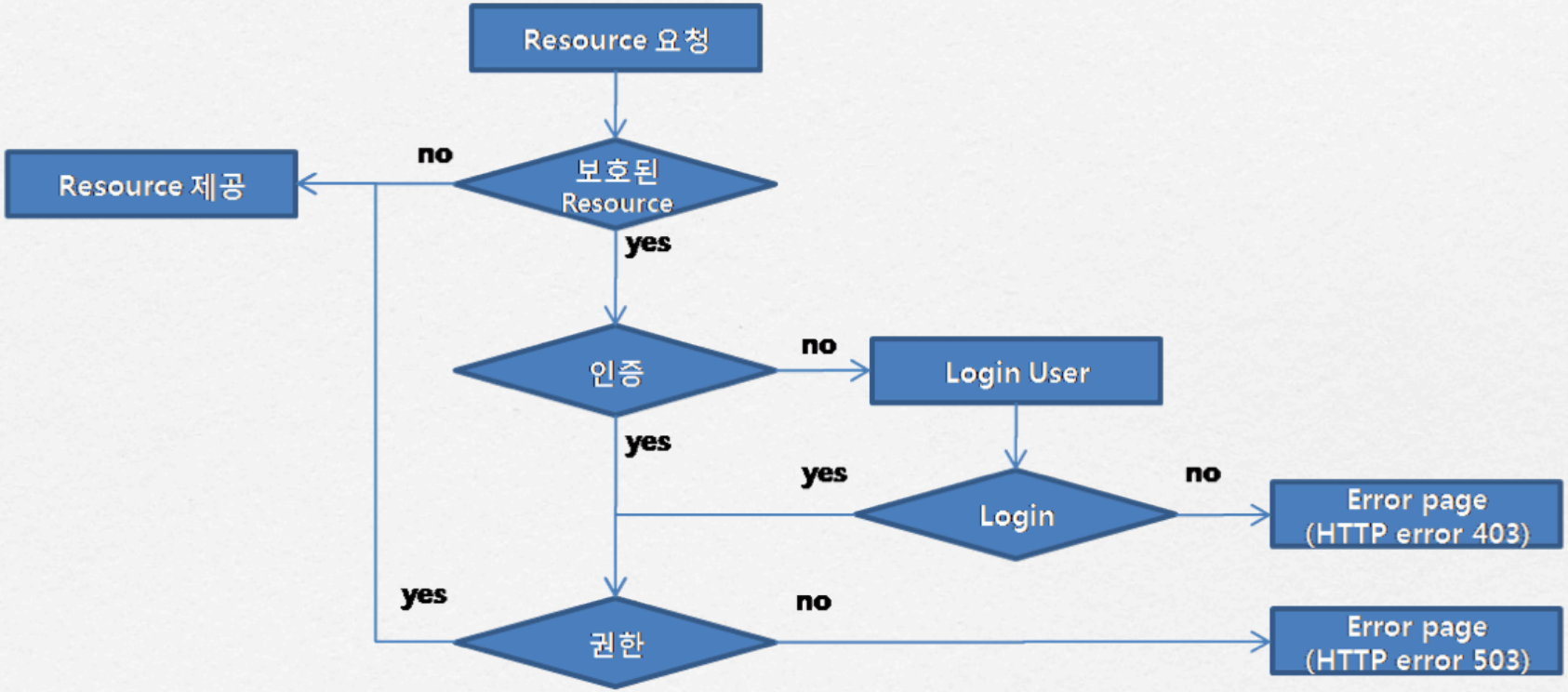
2012년 4월 25일

목차

I. 개요 - security?

- Security : 인증(authentication) + 권한확인(authorization or access-control)
- authentication : 해당 사용자가 본인이 맞는지를 확인하는 절차
  - ex : 아이디 & 패스워드 인증, 인증서로그인 등
- authorization : 인증된 사용자(principal)가 요청된 자원을 접근할 수 있는지 결정하는 절차
  - ex : 관리자 페이지 등

I. 개요 - 일반적인 웹프로그램 인증절차



I. 개요 - Spring security

- 엔터프라이즈 어플리케이션을 위한 인증(Authentication), 권한 처리(Authorization) 서비스를 제공하는 강력하고 유연한 보안 솔루션
- Servlet Filter 와 Java AOP 를 통한 Interception를 사용하여 보안을 강제하며 Spring의 IoC 와 lifecycle 서비스 기반으로 동작
- Authentication, Web URL authorization, Method 호출 authorization, 채널 보안(https 강제) 등의 주요기능 제공
- Service Layer 보안 제공으로 Layering issue 해결 및 웹 클라이언트 외의 다양한 rich 클라이언트 / 웹 서비스에 대한 보안 제어 지원
- 재사용성, 이식성, 코드 품질 및 다양한 타 프레임워크 지원

II. 기본설정 - maven 설정

- Dependency 추가

```
<!-- Spring Security -->
<dependency>
   <groupId>org.springframework.security</groupId>
   <artifactId>spring-security-core</artifactId>
   <version>${spring.maven.artifact.version}</version>
</dependency>

<dependency>
   <groupId>org.springframework.security</groupId>
   <artifactId>spring-security-web</artifactId>
   <version>${spring.maven.artifact.version}</version>
</dependency>

<dependency>
   <groupId>org.springframework.security</groupId>
   <artifactId>spring-security-config</artifactId>
   <version>${spring.maven.artifact.version}</version>
</dependency>
```

II. 기본설정 - maven 설정

- Dependency 추가 (egovframework.rte.fdl.security)

```
<!-- Security -->
<dependency>
    <groupId>egovframework.rte</groupId>
    <artifactId>egovframework.rte.fdl.security</artifactId>
    <version>2.0.0</version>
</dependency>
```

```
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-core</artifactId>
    <version>2.0.4</version>
</dependency>

<dependency>
    <groupId>org.springframework.security</groupId>
     <artifactId>spring-security-taglibs</artifactId>
     <version>2.0.4</version>
</dependency>
```

```
<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-acl</artifactId>
    <version>2.0.4</version>
</dependency>

<dependency>
    <groupId>org.springframework.security</groupId>
    <artifactId>spring-security-core-tiger</artifactId>
    <version>2.0.4</version>
</dependency>
```

II. 기본설정 - security filter 추가

- web.xml

```
<context-param>
    <param-name>contextConfigLocation</param-name>
    <param-value>classpath*:egovframework/spring/context-*.xml</param-value>
</context-param>
<listener>
    <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>
</listener>

<servlet>
    <servlet-name>action</servlet-name>
    <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
    ...
    <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
    ...
</servlet-mapping>

<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>
```

II. 기본설정 - DelegatingFilterProxy

- Spring security가 모든 request를 감싸게 해서 강제적으로 보안이 적용되도록 하는 servlet filter
- 실제로는 <filter-name/>에 지정된 이름을 갖는 Spring bean(filter interface 구현)을 호출하는 역할을 담당
  - springSecurityFilterChain : 이후 설정될 Spring security에 의해 자동으로 등록되는 filter bean

```
public void doFilter(ServletRequest request, ServletResponse response, FilterChain filterChain)
        throws ServletException, IOException {

    // Lazily initialize the delegate if necessary.
    if (this.delegate == null) {
        WebApplicationCont
        }
    }

    //
    thi
}
```

```
protected Fil
    Filter d
    if (isTa
        del
    }
    return d
}
```

```
/**
 * Set the name of the target bean in the Spring application context.
 * The target bean must implement the standard Servlet 2.3 Filter interface.
 * <p>By default, the <code>filter-name</code> as specified for the
 * DelegatingFilterProxy in <code>web.xml</code> will be used.
 */
public void setTargetBeanName(String targetBeanName) {
    this.targetBeanName = targetBeanName;
}
```

II. 기본설정 - Sample 프로젝트

Demo

Sample 프로그램 생성 및 기본 설정

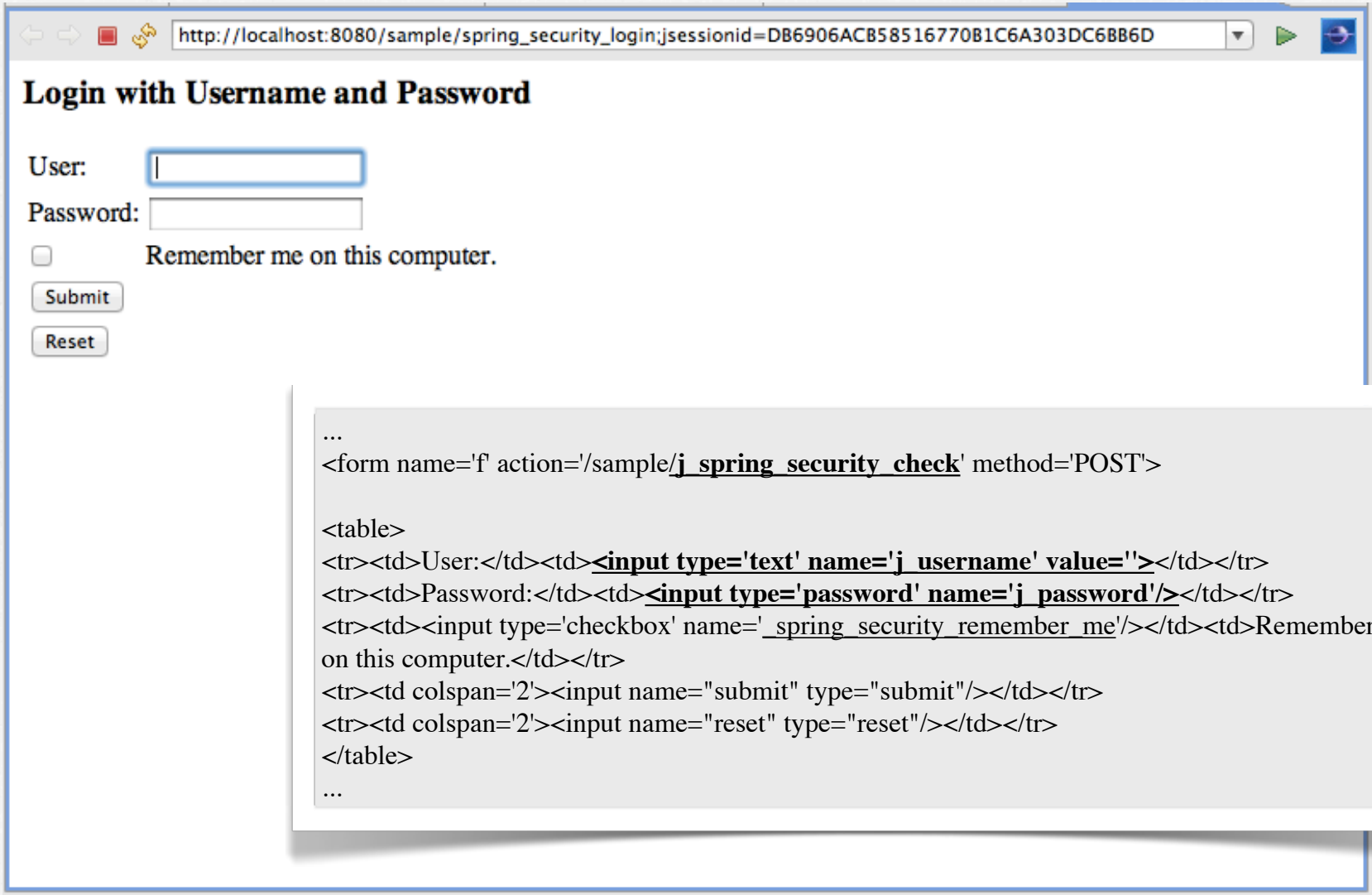
II. 기본설정 - security 설정

- context-security.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans:beans xmlns="http://www.springframework.org/schema/security"
    xmlns:beans="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
    http://www.springframework.org/schema/beans/spring-beans-3.0.xsd
    http://www.springframework.org/schema/security
    http://www.springframework.org/schema/security/spring-security-2.0.4.xsd">

    <http auto-config="true">
        <intercept-url pattern="/sample/add*" access="ROLE_ADMIN" />
        <intercept-url pattern="/sample/update*" access="ROLE_ADMIN" />
        <intercept-url pattern="/sample/delete*" access="ROLE_ADMIN" />
        <intercept-url pattern="/**" access="ROLE_USER" />
    </http>

    <authentication-provider>
        <user-service>
            <user name="user" password="user" authorities="ROLE_USER" />
            <user name="admin" password="admin" authorities="ROLE_USER, ROLE_ADMIN" />
        </user-service>
    </authentication-provider>

</beans:beans>
```

In-memory authentication

II. 기본설정 - 기본 로그인 form




```
...
<form name='f' action='/sample/j_spring_security_check' method='POST'>

<table>
<tr><td>User:</td><td><input type='text' name='j_username' value=''></td></tr>
<tr><td>Password:</td><td><input type='password' name='j_password'/></td></tr>
<tr><td><input type='checkbox' name='_spring_security_remember_me'/></td><td>Remember me
on this computer.</td></tr>
<tr><td colspan='2'><input name="submit" type="submit"/></td></tr>
<tr><td colspan='2'><input name="reset" type="reset"/></td></tr>
</table>
...
```

II. 기본설정 - default form login

Demo

기본 설정 및 Form 로그인 화면

III. Form & Basic 로그인 - auto-config

- auto-config="true"

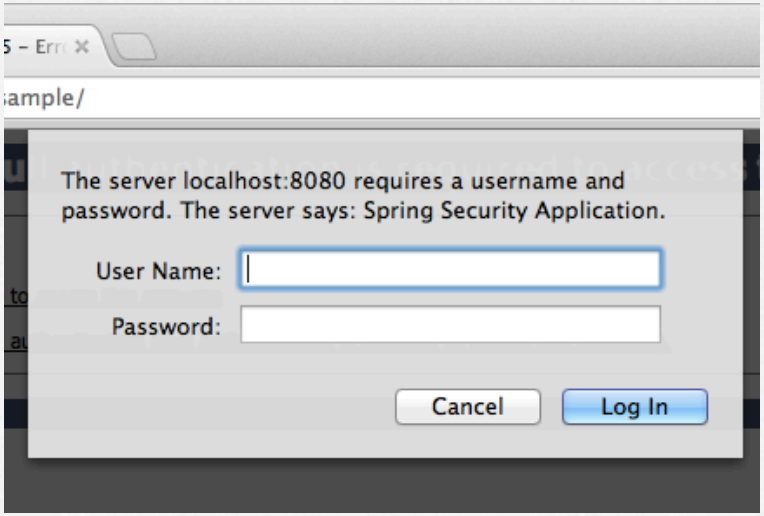
```
<http>
    <form-login />
    <logout />
</http>
```

  - 〈form-login /〉 : /j_spring_security_check
  - 〈logout /〉 : /j_spring_security_logout

III. Form & Basic 로그인 - basic login

- 〈http /〉 설정 변경

```
<http auto-config="true">
    <intercept-url pattern="/sample/add*" access="ROLE_ADMIN" />
    <intercept-url pattern="/sample/update*" access="ROLE_ADMIN" />
    <intercept-url pattern="/sample/delete*" access="ROLE_ADMIN" />
    <intercept-url pattern="/**" access="ROLE_USER" />

    <http-basic />
</http>
```



III. Form & Basic 로그인 - Basic login

Demo

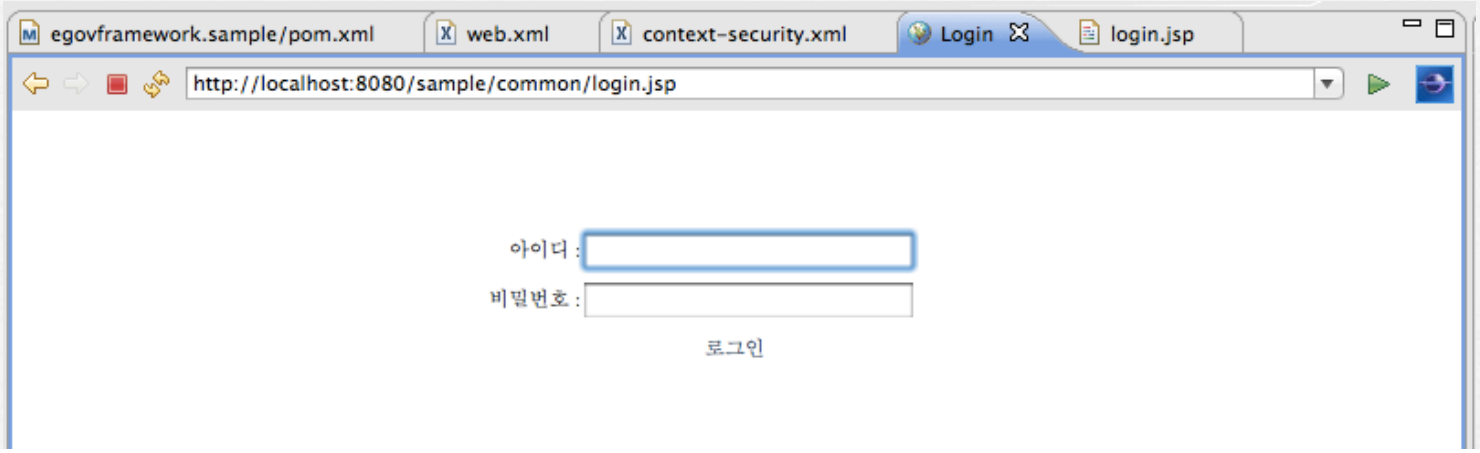
Basic 로그인

III. Form & Basic 로그인 - 화면 변경

- 〈http /〉 설정 변경

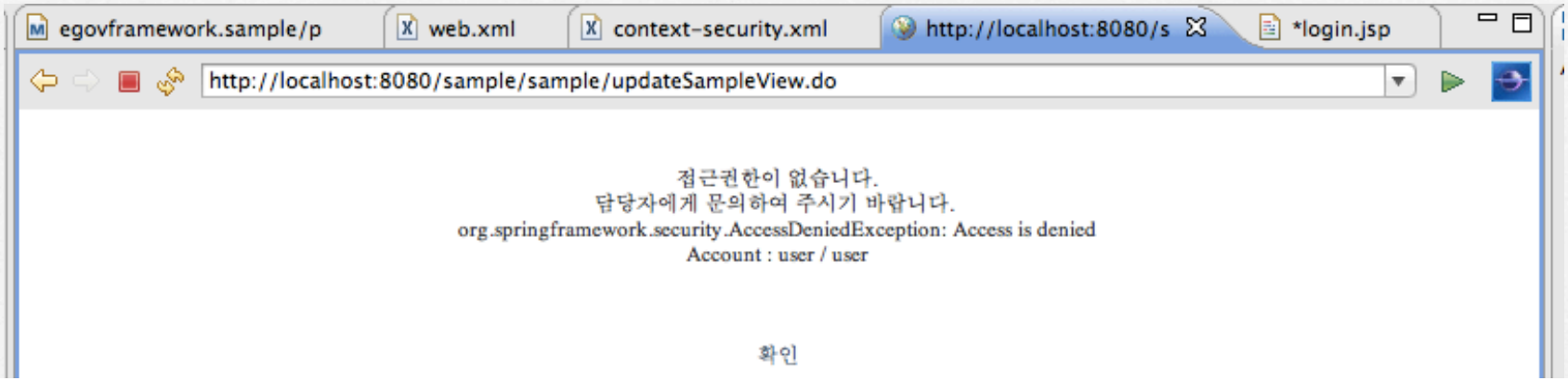
```
<http access-denied-page="/common/accessDenied.jsp" lowercase-comparisons="false">

    <intercept-url pattern="/common/**" access="IS_AUTHENTICATED_ANONYMOUSLY" />
    <intercept-url pattern="/css/**" filters="none" />
    <intercept-url pattern="/images/**" filters="none" />

    <intercept-url pattern="/sample/add*" access="ROLE_ADMIN" />
    <intercept-url pattern="/sample/update*" access="ROLE_ADMIN" />
    <intercept-url pattern="/sample/delete*" access="ROLE_ADMIN" />
    <intercept-url pattern="/**" access="ROLE_USER" />

    <form-login login-page="/common/login.jsp"
                authentication-failure-url="/common/login.jsp?fail=true" />
    <logout logout-success-url="/common/logout.jsp" />
    <anonymous />
</http>
```

III. Form & Basic 로그인 - 로그인 화면




```
<form name="loginForm"  action="<c:url value='/j_spring_security_check'/>">
...
     <c:if test="${not empty param.fail}">
     <font color="red">
          Your login attempt was not successful, try again.<br><br>
          Reason:
     <%= ((AuthenticationException)session.getAttribute(AbstractProcessingFilter.SPRING_SECURITY_LAST_EXCEPTION_KEY)).getMessage() %>
     </font>
     </c:if>
...
     <input type='text' name='j_username'
          <c:if test="${not empty param.fail}">value='<c:out value="${SPRING_SECURITY_LAST_USERNAME}"/>'</c:if> ...

     <input type="password" name="j_password" ...
...
```

III. Form & Basic 로그인 - 권한없음 화면




```
<%@ page import="org.springframework.security.context.SecurityContextHolder" %>
<%@ page import="org.springframework.security.Authentication" %>
<%@ page import="org.springframework.security.ui.AccessDeniedHandlerImpl" %>
<%@ page import="org.springframework.security.userdetails.UserDetails" %>
<%@ page import="org.springframework.security.userdetails.UserDetailsService" %>
...
접근권한이 없습니다.<br> 담당자에게 문의하여 주시기 바랍니다. <br>
<%= request.getAttribute(AccessDeniedHandlerImpl.SPRING_SECURITY_ACCESS_DENIED_EXCEPTION_KEY)%><br>
<%
      Authentication auth = SecurityContextHolder.getContext().getAuthentication();
      Object principal = auth.getPrincipal();
      if (principal instanceof UserDetails) {
            String username = ((UserDetails) principal).getUsername();
            String password = ((UserDetails) principal).getPassword();
            out.println("Account : " + username.toString() + " / " + password.toString() + "<br>");
      }
%>
...
```

III. Form & Basic 로그인 - 화면 변경

Demo

화면 커스터마이징

III. Form & Basic 로그인 - default target

- 〈http /〉 설정 변경

```
<form-login login-page="/common/login.jsp"
      authentication-failure-url="/common/login.jsp?fail=true"
      default-target-url="/home.jsp"
      always-use-default-target='true' />
```

III. Form & Basic 로그인 - password

- Password Encoder 사용

```
<authentication-provider>
    <user-service>
        <user name="user" password="04f8996da763b7a969b1028ee30..."
            authorities="ROLE_USER" />
         <user name="admin" password="8c6976e5b5410415bde908bd4..."
            authorities="ROLE_USER, ROLE_ADMIN" />
      </user-service>

      <password-encoder hash="sha-256"></password-encoder>

</authentication-provider>
```

III. Form & Basic 로그인 - password

- Password 생성하기

```
import o.s.s.providers.encoding.PasswordEncoder;
import o.s.s.providers.encoding.ShaPasswordEncoder;

...

PasswordEncoder encoder = new ShaPasswordEncoder(256);
String hashed = encoder.encodePassword(password, null);

System.out.println("Password : " + password + " => " + hashed);
```

IV. Database 확장 - 기본 확장

- JDBC user service

```
<authentication-provider user-service-ref="jdbcUserService">
      <password-encoder hash="sha-256" />
</authentication-provider>

<jdbc-user-service id="jdbcUserService" data-source-ref="dataSource"/>
```

- 기본 테이블
  - users, authorities
  - groups, groups_authorities, group_members

query는
o.s.s.userdetails.jdbc.JdbcUserDetailsManager에서 확인 가능

IV. Database 확장 - 기본 schema

- Schema (in spring reference appendix)

```
create table users(
  username varchar_ignorecase(50) not null primary key,
  password varchar_ignorecase(50) not null,
  enabled boolean not null);

create table authorities (
  username varchar_ignorecase(50) not null,
  authority varchar_ignorecase(50) not null,
  constraint fk_authorities_users
  foreign key(username) references users(username));

create unique index ix_auth_username
  on authorities (username,authority);
```

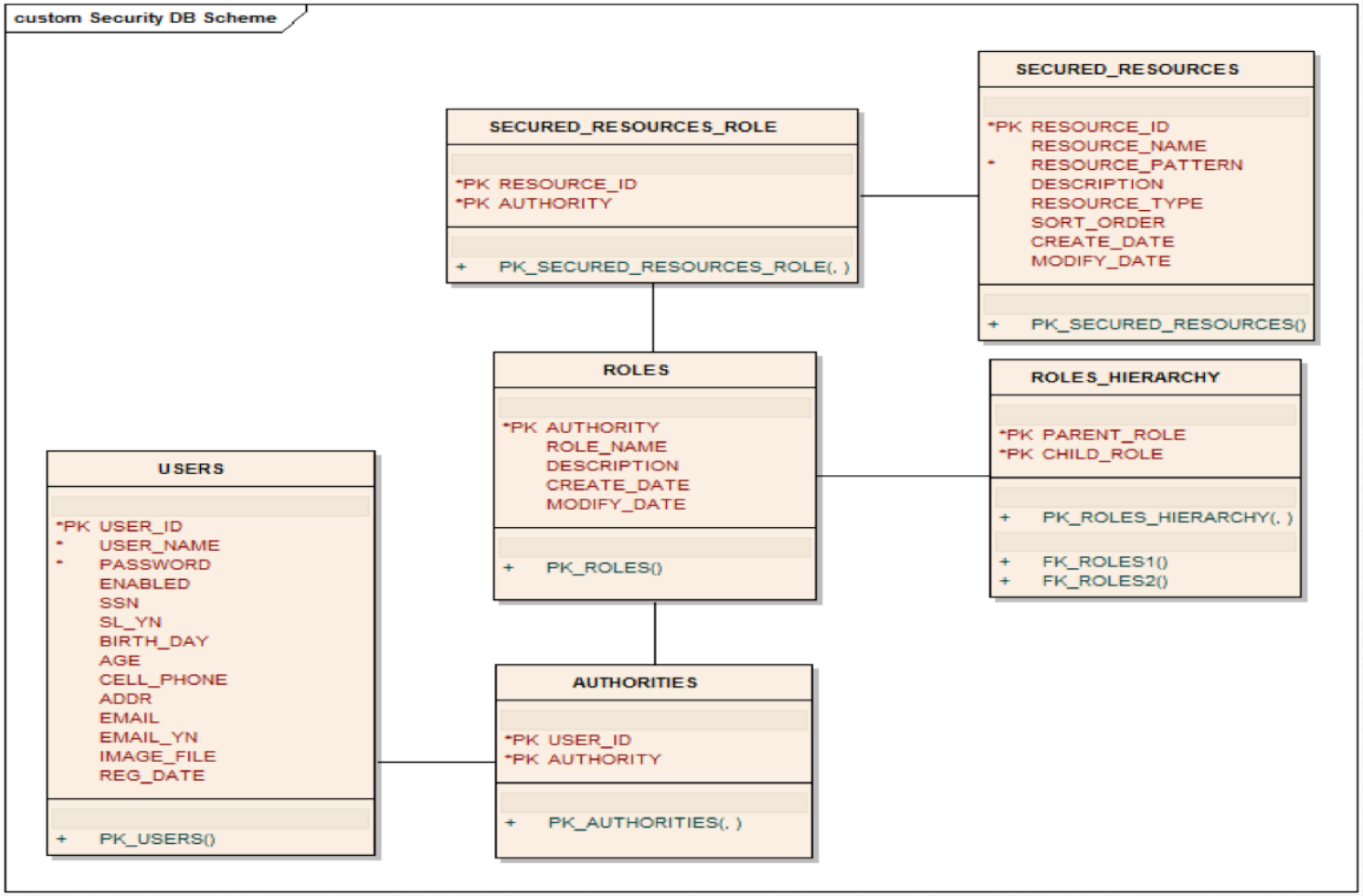
IV. Database 확장 - 기본 schema 변경

- jdbc-user-service의 property 변경

```
<jdbc-user-service id="jdbcUserService"
      data-source-ref="dataSource"
   users-by-username-query="SELECT USER_ID, PASSWORD, ENABLED,
                            BIRTHDAY FROM USERS WHERE USER_ID = ?"
   authorities-by-username-query="SELECT USER_ID, AUTHORITY
                            FROM AUTHORITIES WHERE USER_ID = ?" />
```

IV. Database 확장 - 표준프레임워크 확장

- Schema 적용 예



IV. Database 확장 - 인증 확장 DB 처리

- Authentication 관련 테이블 (1/2)

```
create table users (
   user_id varchar(20) not null primary key,
   password varchar(50) not null,
   enabled boolean not null,
   user_name varchar(50) not null
);

create table roles (
   authority varchar(50) not null primary key,
   role_name varchar(50)
);
```

IV. Database 확장 - 인증 확장 DB 처리 (cont.)

- Authentication 관련 테이블 (2/2)

```
create table authorities (
   user_id varchar(20) not null,
   authority varchar(50) not null,
   primary key(user_id, authority),
   foreign key(user_id) references users(user_id),
   foreign key(authority) references roles(authority)
);

create table roles_hierarchy (
   parent_role varchar(50) not null,
   child_role varchar(50) not null,
   primary key(parent_role, child_role),
   foreign key(parent_role) references roles(authority),
   foreign key(child_role) references roles(authority)
);
```

IV. Database 확장 - 인증 확장 DB 처리 (cont.)

- Authentication 관련 데이터

```
Insert into USERS (USER_ID,USER_NAME,PASSWORD,ENABLED) values
('admin','Admin.','8c6976e5b5410415bde908bd4dee15dfb167a9c873fc4bb8a81f6f2ab448a918',true);
Insert into USERS (USER_ID,USER_NAME,PASSWORD,ENABLED) values
('user','User','04f8996da763b7a969b1028ee3007569eaf3a635486ddab211d512c85b9df8fb',true);

Insert into ROLES (AUTHORITY,ROLE_NAME) values ('ROLE_ADMIN','Administrator Role');
Insert into ROLES (AUTHORITY,ROLE_NAME) values ('ROLE_USER','Public user');
Insert into ROLES (AUTHORITY,ROLE_NAME) values ('ROLE_RESTRICTED','Restricted user');

Insert into AUTHORITIES (USER_ID,AUTHORITY) values ('admin','ROLE_ADMIN');
Insert into AUTHORITIES (USER_ID,AUTHORITY) values ('user','ROLE_USER');

Insert into ROLES_HIERARCHY (PARENT_ROLE,CHILD_ROLE) values ('ROLE_RESTRICTED','ROLE_USER');
Insert into ROLES_HIERARCHY (PARENT_ROLE,CHILD_ROLE) values ('ROLE_USER','ROLE_ADMIN');
```

IV. Database 확장 - 인증 확장 설정

- EgovJdbcUserDetailsManager 적용

```
<!-- customizing user table, authorities table -->
<beans:bean id="jdbcUserService"
    class="egovframework.rte.fdl.security.userdetails.jdbc.EgovJdbcUserDetailsManager">
  <beans:property name="usersByUsernameQuery"
       value="SELECT USER_ID,PASSWORD,ENABLED,USER_NAME FROM USERS WHERE USER_ID = ? "/>
  <beans:property name="authoritiesByUsernameQuery"
       value="SELECT USER_ID,AUTHORITY FROM AUTHORITIES WHERE USER_ID = ? "/>
  <beans:property name="roleHierarchy" ref="roleHierarchy" />
  <beans:property name="dataSource" ref="dataSource" />
  <beans:property name="mapClass"
       value="egovframework.rte.cmmn.security.EgovUserDetailsMapping" />
</beans:bean>
```

- 기존 <jdbc-user-service /> 와의 차이점
  (o.s.s.userdetails.jdbc.JdbcUserDetailsManager extends JdbcDaoImpl)
  - Role Hierarchy 지원 (기존의 경우 별도 UserDetailsServiceWrapper 또는 RoleHierarchyVoter 필요)
  - Mapping class(MappingSqlQuery) 지원 (기존의 경우 상속 필요)

IV. Database 확장 - 인증 확장 설정 (cont.)

- Role hierarchy 적용 (1/2)

```
<beans:bean id="roleHierarchy"
  class="o.s.s.userdetails.hierarchicalroles.RoleHierarchyImpl">
        <!-- XML 사용
        <beans:property name="hierarchy">
            <beans:value>
                ROLE_ADMIN > ROLE_USER
                ROLE_USER > ROLE_RESTRICTED
                ROLE_RESTRICTED > IS_AUTHENTICATED_FULLY
                IS_AUTHENTICATED_REMEMBERED > IS_AUTHENTICATED_ANONYMOUSLY
            </beans:value>
        </beans:property> -->
        <!-- DB 사용 -->
    <beans:property name="hierarchy" ref="hierarchyStrings"/>
</beans:bean>

<beans:bean id="hierarchyStrings"
    class="e.r.fdl.security.userdetails.hierarchicalroles.HierarchyStringsFactoryBean"
    init-method="init">
    <beans:property name="securedObjectService" ref="securedObjectService"/>
</beans:bean>
```

IV. Database 확장 - 인증 확장 설정 (cont.)

- Role hierarchy 적용 (2/2)

```
<beans:bean id="securedObjectService"
    class="e.r.fdl.security.securedobject.impl.SecuredObjectServiceImpl">
    <beans:property name="securedObjectDAO" ref="securedObjectDAO"/>
</beans:bean>

<beans:bean id="securedObjectDAO"
    class="e.r.fdl.security.securedobject.impl.SecuredObjectDAO">
    <beans:property name="dataSource" ref="dataSource"/>
</beans:bean>
```

```
public interface EgovSecuredObjectService {
    /** 롤에 대한 URL의 매핑 정보를 얻어온다. */
    public LinkedHashMap getRolesAndUrl() throws Exception;

    /** 롤에 대한 메소드의 매핑 정보를 얻어온다. */
    public LinkedHashMap getRolesAndMethod() throws Exception;

    /** 롤에 대한 AOP pointcut 메핑 정보를 얻어온다. */
    public LinkedHashMap getRolesAndPointcut() throws Exception;

    /** 롤의 계층적 구조를 얻어온다. */
    public String getHierarchicalRoles() throws Exception;
}
```

IV. Database 확장 - 인증 확장 설정 (cont.)

- Map class

```
@Override
protected Object mapRow(ResultSet rs, int rownum) throws SQLException {
    String userid = rs.getString("user_id");
    String password = rs.getString("password");
    boolean enabled = rs.getBoolean("enabled");

    String username = rs.getString("user_name");

    // TODO USERS 테이블 컬럼 변경 (세션에서 관리되는 항목 추가)
    // 예) String birthDay = rs.getString("birth_day");

    EgovUserDetailsVO userVO = new EgovUserDetailsVO();
    userVO.setUserId(userid);
    userVO.setPassWord(password);
    userVO.setUserName(username);

    // TODO USERS 테이블 컬럼 변경 (세션에서 관리되는 항목 추가)
    // 예) userVO.setBirthDay(birthDay);

    return new EgovUserDetails(userid, password, enabled, userVO);
}
```

IV. Database 확장 - 인증 확장

Demo

DB를 통한 확장 인증

IV. Database 확장 - 인증 확장 설정 (cont.)

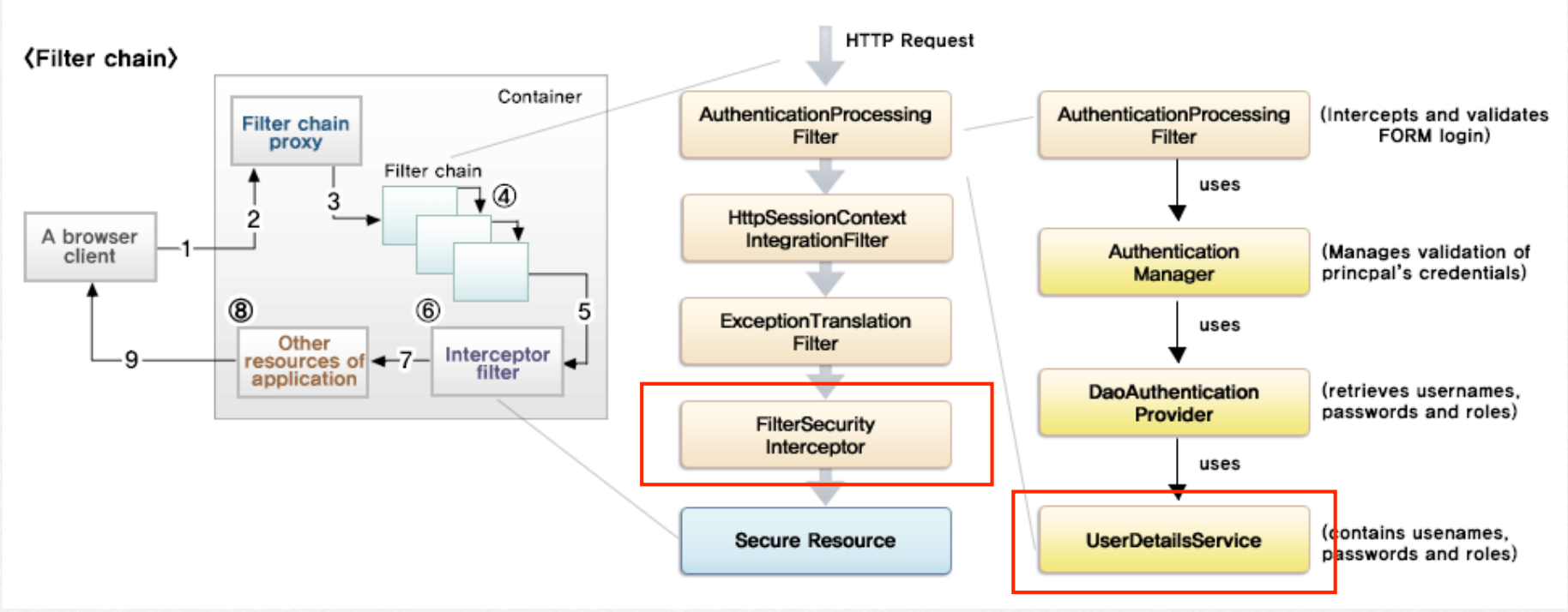
- 로그인 정보 얻기

```
import e.r.fdl.security.userdetails.util.EgovUserDetailsHelper;
...
if (EgovUserDetailsHelper.isAuthenticated()) {
   EgovUserDetailsVO user =
   (EgovUserDetailsVO)EgovUserDetailsHelper.getAuthenticatedUser();

   System.out.println(user.getUserId());
   System.out.println(user.getUserName());

   List<String> autorities = EgovUserDetailsHelper.getAuthorities();
}
```

```
public static Object getAuthenticatedUser() {
    SecurityContext context = SecurityContextHolder.getContext();
    Authentication authentication = context.getAuthentication();

    EgovUserDetails details = (EgovUserDetails) authentication.getPrincipal();

    return details.getEgovUserVO();
}
```

IV. Database 확장 - 권한 확장 개요

- filter chain



from AnyFrame

IV. Database 확장 - 권한 확장 DB 처리

- Authorization 관련 테이블

```
create table secured_resources (
   resource_id varchar(10) not null primary key,
   resource_name varchar(50),
   resource_pattern varchar(100) not null,
   resource_type varchar(10),
   sort_order integer
);

create table secured_resources_role (
   resource_id varchar(10) not null,
   authority varchar(50) not null,
   primary key(resource_id, authority),
   foreign key(resource_id) references secured_resources(resource_id),
   foreign key(authority) references roles(authority)
);
```

IV. Database 확장 - 권한 확장 DB 처리 (cont.)

- Authorization 관련 데이터

```
Insert into SECURED_RESOURCES
(RESOURCE_ID,RESOURCE_NAME,RESOURCE_PATTERN,RESOURCE_TYPE,SORT_ORDER)
values ('WEB-0000001','Sample Add','/sample/add*','url',1);
Insert into SECURED_RESOURCES
(RESOURCE_ID,RESOURCE_NAME,RESOURCE_PATTERN,RESOURCE_TYPE,SORT_ORDER)
values ('WEB-0000002','Sample Update','/sample/update*','url',1);
Insert into SECURED_RESOURCES
(RESOURCE_ID,RESOURCE_NAME,RESOURCE_PATTERN,RESOURCE_TYPE,SORT_ORDER)
values ('WEB-0000003','Sample Delete','/sample/delete*','url',1);
Insert into SECURED_RESOURCES
(RESOURCE_ID,RESOURCE_NAME,RESOURCE_PATTERN,RESOURCE_TYPE,SORT_ORDER)
values ('WEB-0000004','Sample Select','/sample/egov*','url',2);

Insert into SECURED_RESOURCES_ROLE (RESOURCE_ID,AUTHORITY) values
('WEB-0000001','ROLE_ADMIN');
Insert into SECURED_RESOURCES_ROLE (RESOURCE_ID,AUTHORITY) values
('WEB-0000002','ROLE_ADMIN');
Insert into SECURED_RESOURCES_ROLE (RESOURCE_ID,AUTHORITY) values
('WEB-0000003','ROLE_ADMIN');
Insert into SECURED_RESOURCES_ROLE (RESOURCE_ID,AUTHORITY) values
('WEB-0000004','ROLE_USER');
```

IV. Database 확장 - 권한 확장 설정

- using custom-filter
  - add own filters
  - or customize standard namespace filter

```
<beans:bean id="myFilter"
   class="com.mycompany.MySpecialAuthenticationFilter">
      <custom-filter position="AUTHENTICATION_PROCESSING_FILTER"/>
</beans:bean>
```

IV. Database 확장 - 권한 확장 설정 (cont.)

- Standard Filter Aliases and Ordering

Alias	Filter class	Namespace 설정
CHANNEL_FILTER	ChannelProcessingFilter	http/intercept-url
CONCURRENT_SESSION_FILTER	ConcurrentSessionFilter	http/concurrent-session-control
SESSION_CONTEXT_INTEGRATION_FILTER	HttpSessionContextIntegrationFilter	http
LOGOUT_FILTER	LogoutFilter	http/logout
AUTHENTICATION_PROCESSING_FILTER	AuthenticationProcessingFilter	http/form-login
BASIC_PROCESSING_FILTER	BasicProcessingFilter	http/basic-login
REMEMBER_ME_FILTER	RememberMeProcessingFilter	http/remember-me
ANONYMOUS_FILTER	AnonymousProcessingFilter	http/anonymous
EXCEPTION_TRANSLATION_FILTER	ExceptionTranslationFilter	http
FILTER_SECURITY_INTERCEPT	FilterSecurityInterceptor	http

☞ 중요 filter만 표시

IV. Database 확장 - 권한 확장 설정 (cont.)

- filterSecurityInterceptor

```
<beans:bean id="filterSecurityInterceptor"
  class="org.springframework.security.intercept.web.FilterSecurityInterceptor">
    <custom-filter before="FILTER_SECURITY_INTERCEPTOR" />
    <beans:property name="authenticationManager" ref="_authenticationManager" />
    <beans:property name="accessDecisionManager" ref="_accessManager" />
    <beans:property name="objectDefinitionSource"
        ref="databaseObjectDefinitionSource" />
</beans:bean>
```

☞ 자동으로 등록되는 bean 이름 등은 Security schema handler인 o.s.s.config.SecurityNamespaceHander 참조

IV. Database 확장 - 권한 확장 설정 (cont.)

- AccessDecisionManager (기본 설정)

```
<beans:bean id="_accessManager"
        class="org.springframework.security.vote.AffirmativeBased">
    <beans:property name="allowIfAllAbstainDecisions" value="false" />
    <beans:property name="decisionVoters">
        <beans:list>
            <beans:bean class="org.springframework.security.vote.RoleVoter">
                <beans:property name="rolePrefix" value="ROLE_" />
            </beans:bean>
            <beans:bean class="org.springframework.security.vote.AuthenticatedVoter" />
        </beans:list>
    </beans:property>
</beans:bean>
```

  - o.s.s.vote.AffirmativeBased : 단 한 개의 조건도 허락하면 허가함
  - o.s.s.vote.UnanimousBased : 모든 조건이 허락해야 허가함
  - o.s.s.vote.RoleVoter : 지정된 role을 갖는 경우 허가됨
  - o.s.s.vote.AuthenticatedVoter : 인증되고 IS_AUTHENTICATED_*를 role을 갖는 경우 허가됨

IV. Database 확장 - 권한 확장 설정 (cont.)

- objectDefinitionSource

```
<beans:bean id="databaseObjectDefinitionSource"
class="o.s.s.intercept.web.EgovReloadableDefaultFilterInvocationDefinitionSource">
    <beans:constructor-arg ref="antUrlPathMatcher" />
    <!--beans:constructor-arg ref="regexUrlPathMatcher" /-->
    <beans:constructor-arg ref="requestMap" />
    <beans:property name="securedObjectService" ref="securedObjectService"/>
</beans:bean>

<!--  url  -->
<beans:bean id="requestMap"
class="e.r.fdl.security.intercept.ResourcesMapFactoryBean"
    init-method="init">
    <beans:property name="securedObjectService" ref="securedObjectService"/>
    <beans:property name="resourceType" value="url"/>
</beans:bean>

<beans:bean id="antUrlPathMatcher" class="o.s.s.util.AntUrlPathMatcher" />
<beans:bean id="regexUrlPathMatcher" class="o.s.s.util.RegexUrlPathMatcher" />
```

IV. Database 확장 - 권한 확장 설정 (cont.)

- EgovReloadableDefaultFilterInvocationDefinitionSource

```
public class EgovReloadableDefaultFilterInvocationDefinitionSource
    extends DefaultFilterInvocationDefinitionSource implements ApplicationContextAware {

    private EgovSecuredObjectService securedObjectService;

    public void setSecuredObjectService(EgovSecuredObjectService securedObjectService) {
        this.securedObjectService = securedObjectService;
    }

    public void reloadRequestMap() throws Exception {
        // ...
    }

    /* requestMap에 대해서 호출되어 Secured URL 정보를 생성 */
    @Override
    void addSecureUrl(String pattern, String method, ConfigAttributeDefinition attr) {
        Map mapToUse = getRequestMap();

        mapToUse.put(getUrlMatcher().compile(pattern), attr);
        //...
    }
}
```

IV. Database 확장 - 권한 확장 설정 (cont.)

- ResourcesMapFactoryBean

```
public class ResourcesMapFactoryBean implements FactoryBean {
    private LinkedHashMap resourcesMap;
    private String resourceType;

    public void init() throws Exception {
        if ("method".equals(resourceType)) {
            resourcesMap = securedObjectService.getRolesAndMethod();
        } else if ("pointcut".equals(resourceType)) {
            resourcesMap = securedObjectService.getRolesAndPointcut();
        } else {
            resourcesMap = securedObjectService.getRolesAndUrl();
        }
    }

    public Object getObject() throws Exception {
        if (resourcesMap == null) {
            init();
        }
        return resourcesMap;
    }
    //...
}
```

IV. Database 확장 - 권한 확장

Demo

DB를 통한 권한확인 확장

V. Method & Pointcut - method 보안

- 〈global-method-security〉

```
<global-method-security secured-annotations="enabled"
   jsr250-annotations="enabled"/>

public interface BankService {
  @Secured("IS_AUTHENTICATED_ANONYMOUSLY")
  public Account readAccount(Long id);
  @Secured("IS_AUTHENTICATED_ANONYMOUSLY")
  public Account[] findAccounts();
  @Secured("ROLE_TELLER")
  public Account post(Account account, double amount);
}

<global-method-security>
   <protect-pointcut expression="execution(* com..*Service.*(..))"
   access="ROLE_USER"/>
</global-method-security>
```

V. Method & Pointcut - method 보안 (cont.)

- 〈intercept-methods〉

```
<beans:bean id="target" class="com.mycompany.myapp.MyBean">
    <intercept-methods>
        <protect method="set*" access="ROLE_ADMIN" />
        <protect method="get*" access="ROLE_ADMIN,ROLE_USER" />
        <protect method="doSomething" access="ROLE_USER" />
    </intercept-methods>
</beans:bean>
```

V. Method & Pointcut - method 보안 (cont.)

- MethodSecurityInterceptor

```
<beans:bean id="methodDefinitionSourceAdvisor"
class="o.s.s.intercept.method.aopalliance.MethodDefinitionSourceAdvisor">
   <beans:constructor-arg value="methodSecurityInterceptor" />
   <beans:constructor-arg ref="delegatingMethodDefinitionSource" />
</beans:bean>

<beans:bean id="methodSecurityInterceptor"
class="o.s.s.intercept.method.aopalliance.MethodSecurityInterceptor">
   <beans:property name="validateConfigAttributes" value="false" />
   <beans:property name="authenticationManager" ref="_authenticationManager"/>
   <beans:property name="accessDecisionManager" ref="_accessManager"/>
   <beans:property name="objectDefinitionSource"
      ref="delegatingMethodDefinitionSource" />
   </beans:bean>
```

  - AdvisorAutoProxyCreator에 의해 생성되는 MethodDefinitionSourceAdvisor Advice를 통해 MethodSecurityInterceptor 호출

V. Method & Pointcut - method 보안 (cont.)

- DelegatingMethodDefinitionSource

```
<beans:bean id="delegatingMethodDefinitionSource"
class="o.s.s.intercept.method.DelegatingMethodDefinitionSource">
   <beans:property name="methodDefinitionSources">
      <beans:list>
         <beans:ref bean="methodDefinitionSources"/>
         <beans:bean
            class="o.s.s.annotation.SecuredMethodDefinitionSource" />
         <beans:bean
            class="o.s.s.annotation.Jsr250MethodDefinitionSource" />
         </beans:list>
      </beans:property>
   </beans:bean>
```

V. Method & Pointcut - method 보안 (cont.)

- MethodDefinitionSource

```
<beans:bean id="methodDefinitionSources"
      class="o.s.s.intercept.method.MapBasedMethodDefinitionSource">
   <beans:constructor-arg ref="methodMap" />
</beans:bean>

<beans:bean id="methodMap"
      class="e.r.fdl.security.intercept.ResourcesMapFactoryBean"
      init-method="init">
   <beans:property name="securedObjectService"
      ref="securedObjectService"/>
   <beans:property name="resourceType" value="method"/>
</beans:bean>
```

V. Method & Pointcut - method 보안

Demo

DB를 통한 메소드 보안 확장

V. Method & Pointcut - pointcut 보안

- MethodDefinitionSource

```
<beans:bean id="protectPointcutPostProcessor"
   class="o.s.s.intercept.method.ProtectPointcutPostProcessor">
   <beans:constructor-arg ref="methodDefinitionSources" />
   <beans:property name="pointcutMap" ref="pointcutMap"/>
</beans:bean>

<beans:bean id="pointcutMap"
   class="e.r.fdl.security.intercept.ResourcesMapFactoryBean"
   init-method="init">
   <beans:property name="securedObjectService"
      ref="securedObjectService"/>
   <beans:property name="resourceType" value="pointcut"/>
</beans:bean>
```

VI. 기타 활용 - https channel 보안

- requires-channel

```
<http>
   <intercept-url pattern="/secure/**" access="ROLE_USER"
      requires-channel="https"/>
   <intercept-url pattern="/**" access="ROLE_USER"
      requires-channel="any"/>
  ...
   <port-mappings>
      <port-mapping http="9080" https="9443"/>
   </port-mappings>
</http>
```

  - https로 지정된 URL을 http로 호출하면 https로 redirect됨

VI. 기타 활용 - 동시접속 제한

- 〈concurrent-session-control〉

```
<http>
   ...
   <concurrent-session-control max-sessions="1"
      exception-if-maximum-exceeded="true"/>
</http>
```

- web.xml 설정 추가

```
<listener>
   <listener-class>o.s.s.ui.session.HttpSessionEventPublisher
   </listener-class>
</listener>
```

☞ HttpSessionListener로 session 생성 시(sessionCreated) 또는 삭제 시(sessionDestroyed) 시 해당 이벤트를 처리

VI. 기타 활용 - taglib 활용

- security tags

```
<%@ taglib prefix='security' uri='http://www.springframework.org/
security/tags' %>
...
<security:authorize ifNotGranted="ROLE_USER">
   <a href="<c:url value='/common/login.jsp' />">login</a>
</security:authorize>
<security:authorize ifAnyGranted="ROLE_USER, ROLE_ADMIN">
   <b><security:authentication property="principal.username" /></b>님
      반갑습니다.<br>
   <a href="<c:url value='/j_spring_security_logout' />">logout</a>
</security:authorize>
```

VI. 기타 활용 - Remember-Me

- Simple Hash-based token 방식

```
<http>
   ...
   <remember-me key="myAppKey"/>
</http>
```

- Persistent token 방식

```
<http>
   ...
   <remember-me data-source-ref="someDataSource"/>
</http>
```

```
create table persistent_logins (
  username varchar(64) not null,
  series varchar(64) primary key,
  token varchar(64) not null,
  last_used timestamp not null);
```

References

- Spring Security Reference Document (http://static.springsource.org/spring-security/site/docs/2.0.x/reference/springsecurity.html)
- Spring Framework Programming (javapassion.com)
- 전자정부 표준프레임워크 실행환경 가이드 (http://www.egovframe.org/wiki/doku.php?id=egovframework:rte2:fdl:server_security)
- AnyFrame IAM (http://www.anyframejava.org/project/iam)

감사합니다.