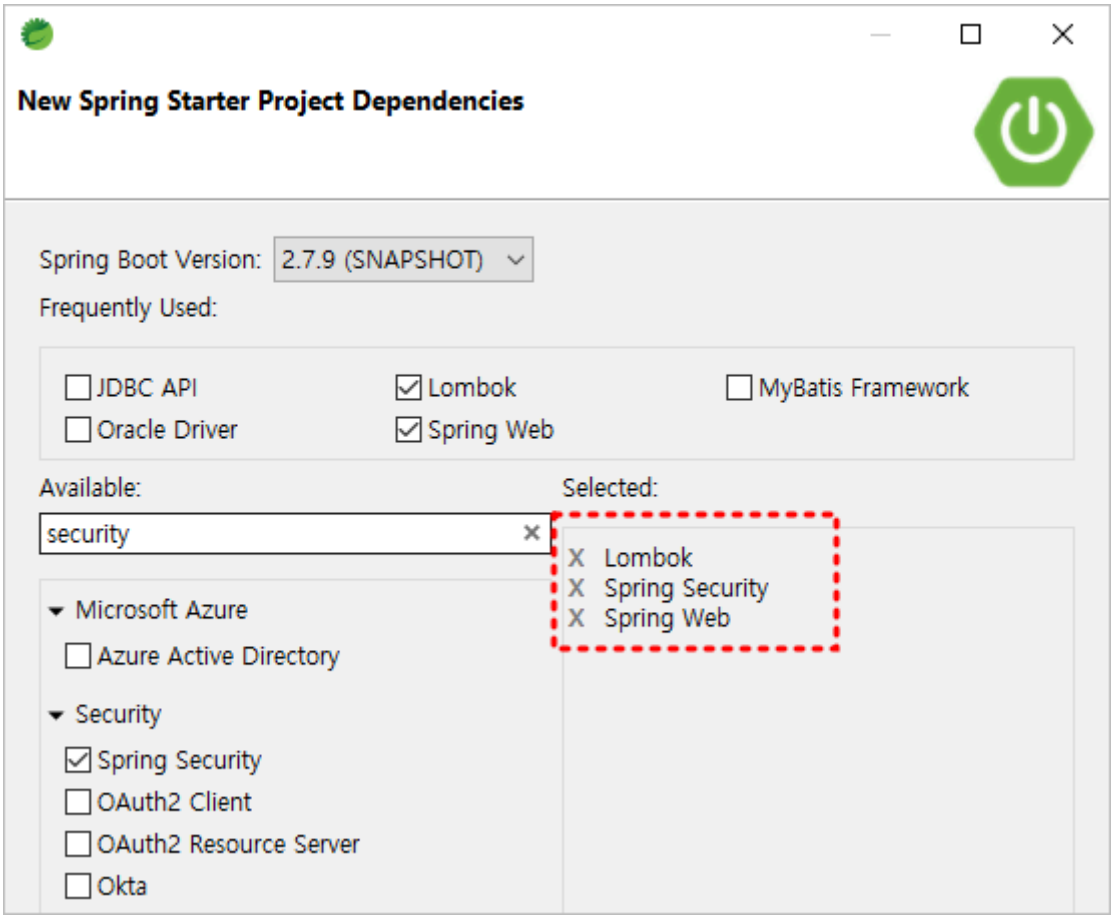
시큐리티 - Security 설정

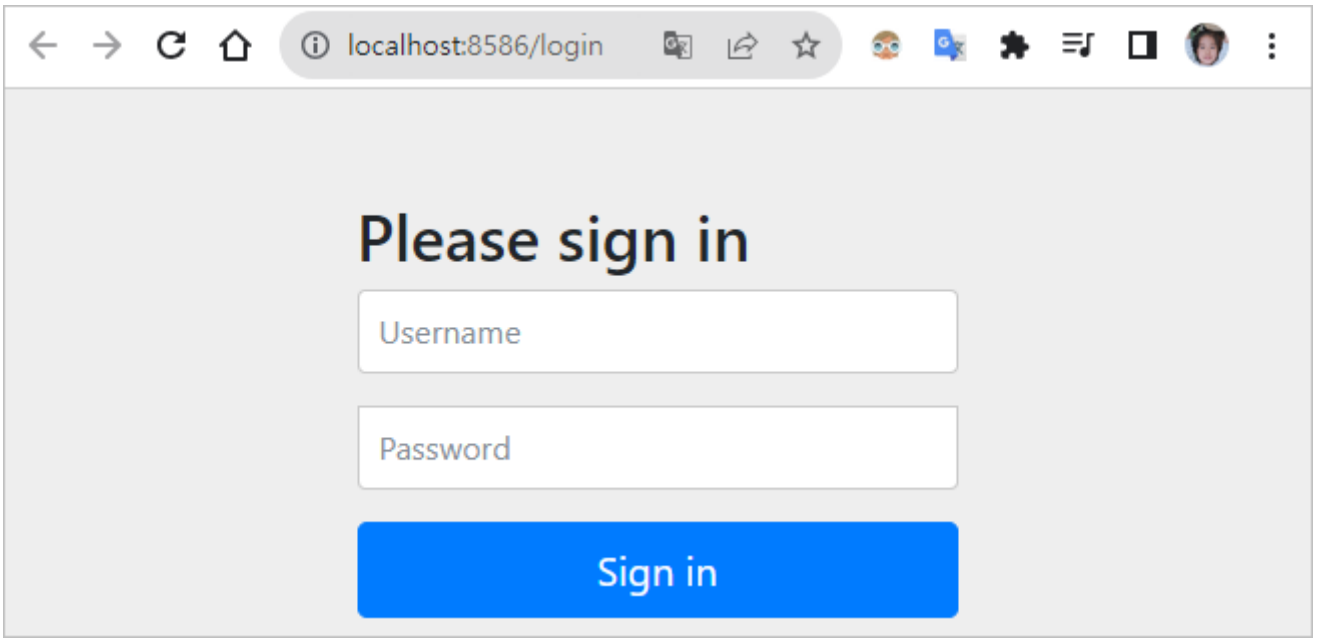
프로젝트명 : B09aSecurity

준비사항

- 의존설정 : Spring Web, Lombok, Spring Security
- JSP 사용을 위한 설정을 한다.
- Refresh Gradle Project 를 눌러 적용한다.



여기까지 설정한 후 실행해보면 다음과 같이 로그인 페이지가 출력된다.

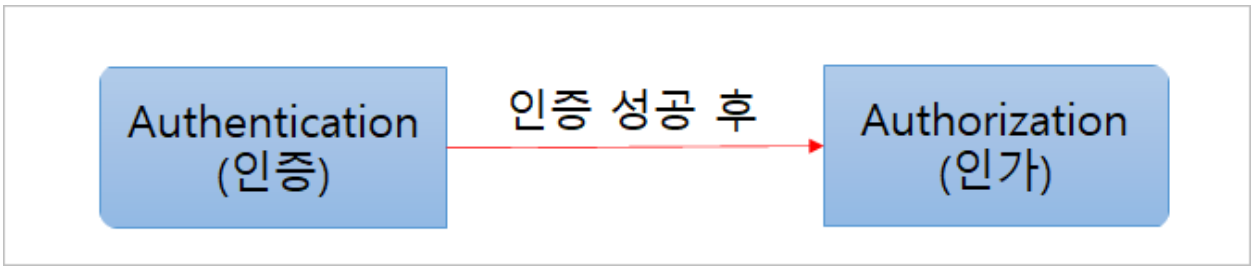


Spring Security란?

- Spring Security는 Spring기반의 **애플리케이션의 보안**(인증, 인가, 권한 등)을 담당하는 스프링 하위 프레임워크이다.
- '인증'과 '권한부여'에 대한 부분을 **Filter** 흐름에 따라 처리한다.
- **Filter**는 **Dispatcher Servlet**으로 가기 전에 적용되므로 가장 먼저 URL 요청을 받지만, **Interceptor**는 **Dispatcher**와 **Controller**사이에 위치한다는 점에서 적용 시기의 차이가 있다.
- 보안과 관련해서 체계적으로 많은 옵션을 제공해주기 때문에 개발자 입장에서는 일일이 보안관련 로직을 작성하지 않아도 된다는 장점이 있다.

인증(Authentication)과 인가(Authorizatoin)

- 인증(Authentication) : 해당 사용자가 본인이 맞는지를 확인하는 절차
- 인가(Authorization) : 인증된 사용자가 요청한 자원에 접근 가능한지를 결정하는 절차



Spring Security는 인증 절차를 거친 후에 인가 절차를 진행하게 되며, 인가 과정에서 해당 리소스에 대한 접근 권한이 있는지 확인한다. Spring Security에서는 이러한 인증과 인가를 위해 Principal을 아이디로, Credential을 비밀번호로 사용하는 **Credential 기반의 인증 방식**을 사용한다.

- Principal(접근 주체) : 보호받는 Resource에 접근하는 대상
- Credential(비밀번호) : Resource에 접근하는 대상의 비밀번호

Security 설정하기

시큐리티 설정 : com.edu.springboot.auth.WebSecurityConfig.java

```java
package com.edu.springboot.auth;

import org.springframework.context.annotation.Bean;

@Configuration
public class WebSecurityConfig {

    @Bean
    public SecurityFilterChain filterChain(HttpSecurity http)
            throws Exception {
        http.csrf((csrf) -> csrf.disable())
            .cors((cors) -> cors.disable())
            .authorizeHttpRequests((request) -> request
                .dispatcherTypeMatchers(DispatcherType.FORWARD).permitAll()
                .requestMatchers("/").permitAll()
                .requestMatchers("/css/**", "/js/**", "/images/**").permitAll()
                .requestMatchers("/guest/**").permitAll()
                .requestMatchers("/member/**").hasAnyRole("USER", "ADMIN")
                .requestMatchers("/admin/**").hasRole("ADMIN")
                .anyRequest().authenticated()
            );

        http.formLogin((formLogin) ->
                formLogin.permitAll());

        http.logout((logout) ->
                logout.permitAll());

        return http.build();
    }
```

```java
    @Bean
    public UserDetailsService users() {
        UserDetails user = User.builder()
                .username("user")
                .password(passwordEncoder().encode("1234"))
                .roles("USER")
                .build();
        UserDetails admin = User.builder()
                .username("admin")
                .password(passwordEncoder().encode("1234"))
                .roles("USER", "ADMIN")
                .build();

        // 메모리에 사용자 정보를 담는다.
        return new InMemoryUserDetailsManager(user, admin);
    }
```

```java
    // 패스워드 인코더(암호화)
    public PasswordEncoder passwordEncoder() {
        return PasswordEncoderFactories.createDelegatingPasswordEncoder();
    }
}
```

컨트롤러 : com.edu.springboot.MainController.java

```java
package com.edu.springboot;

import org.springframework.stereotype.Controller;

@Controller
public class MainController {

    @RequestMapping("/")
    public String home() {
        return "home";
    }

    @RequestMapping("/guest/index.do")
    public String welcome1() {
        return "guest";
    }

    @RequestMapping("/member/index.do")
    public String welcome2() {
        return "member";
    }

    @RequestMapping("/admin/index.do")
    public String welcome3() {
        return "admin";
    }
}
```

여기까지 작성하세요.

정적리소스(이미지) : resources/static/images/security.jpg [다운로드]

정적리소스(CSS) : resources/static/css/main.css

```css
*{
    font-size: 12px;font-family: verdana;
}
.spring{
    width: 100%; height: 187px;
    background: url('../images/security.jpg') repeat;
}

```

정적리소스(Javascript) : resources/static/js/commons.js

```javascript
function myAlert(){
    alert("경고창이 표시됩니다.")
}

let myConsole = function(param){
    console.log("파라미터:"+ param);
}

```

홈 : webapp/WEB-INF/views/home.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Spring Boot Project</title>
<script src="./js/commons.js"></script>
<link rel="stylesheet" href="./css/main.css" />
</head>
<body>
    <h2>스프링 부트 프로젝트</h2>
    <ul>
        <li><a href="/">루트</a></li>
    </ul>

    <script>
        window.onload = function(){
            myConsole("시큐리티");
        }
    </script>

```

```
23      <h2>Spring Security 기본</h2>
24      <ul>
25          <li><a href="/guest/index.do">Guest</a></li>
26          <li><a href="/member/index.do">Member</a></li>
27          <li><a href="/admin/index.do">Admin</a></li>
28      </ul>
29
30      <div class="spring" onclick="myAlert();">
31          여기를 클릭해보세요.
32      </div>
33  </body>
34  </html>
```

뷰 : webapp/WEB-INF/views/admin.jsp

```
 9  <body>
10      <h2>Admin영역</h2>
11      ADMIN권한만 접근할 수 있습니다.
12      <%@ include file="/link.jsp" %>
13  </body>
```

뷰 : webapp/WEB-INF/views/member.jsp

```
 9  <body>
10      <h2>Member영역</h2>
11      ADMIN or USER권한이 있어야 접근할 수 있습니다.
12      <%@ include file="/link.jsp" %>
13  </body>
```

뷰 : webapp/WEB-INF/views/guest.jsp

```
 9  <body>
10      <h2>Guest영역</h2>
11      권한 없이 누구나 접근할 수 있습니다.
12      <%@ include file="/link.jsp" %>
13  </body>
```

뷰 : webapp/link.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<h2>바로가기</h2>
<ul>
    <li><a href="/login">Default Login</a></li>
    <li><a href="/guest/index.do">Guest</a></li>
    <li><a href="/member/index.do">Member</a></li>
    <li><a href="/admin/index.do">Admin</a></li>
    <li><a href="/logout">Default Logout</a></li>
</ul>

```

각 링크를 눌러 접속해본다.

현재는 user로 로그인 한 후 Admin으로 접속하면 “Whitelabel Error Page”가 발생한다.