(/)

How to Manually Authenticate User with Spring Security

FEATURED VIDEOS

Spring Security - Remember Me w...
LEARN SPRING SECURITY
Remember Me
Remember Me with Persistence

Last updated: January 8, 2024

Written by: baeldung (https://www.baeldung.com/author/baeldung)

Reviewed by: Grzegorz Piwowarek (https://www.baeldung.com/author/grzegorz-author)

Spring Security (https://www.baeldung.com/category/spring/spring-security)

Authentication (https://www.baeldung.com/tag/authentication)

1. Overview

In this quick article, we'll focus on how to programmatically set an authenticated user in Spring Security and Spring MVC.

2. Spring Security

Simply put, Spring Security hold the principal information of each authenticated user in a *ThreadLocal* – represented as an *Authentication* object.

In order to construct and set this *Authentication* object – we need to use the same approach Spring Security typically uses to build the object on a standard authentication.

To, let's manually trigger authentication and then set the resulting *Authentication* object into the current *SecurityContext* used by the framework to hold the currently logged-in user:

```
UsernamePasswordAuthenticationToken authReq
 = new UsernamePasswordAuthenticationToken(user, pass);
Authentication auth = authManager.authenticate(authReq);
SecurityContext sc = SecurityContextHolder.getContext();
sc.setAuthentication(auth);
```

After setting the *Authentication* in the context, we'll now be able to check if the current user is authenticated – using *securityContext.getAuthentication().isAuthenticated()*.

3. Spring MVC

By default, Spring Security adds an additional filter in the Spring Security filter chain – which is capable of persisting the Security Context (*SecurityContextPersistenceFilter* class).

In turn, it delegates the persistence of the Security Context to an instance of *SecurityContextRepository*, defaulting to the *HttpSessionSecurityContextRepository* class.

So, in order to set the authentication on the request and hence, **make it available for all subsequent requests from the client**, we need to manually set the *SecurityContext* containing the *Authentication* in the HTTP session:

```
public void login(HttpServletRequest req, String user, String pass) {
    UsernamePasswordAuthenticationToken authReq
      = new UsernamePasswordAuthenticationToken(user, pass);
    Authentication auth = authManager.authenticate(authReq);

    SecurityContext sc = SecurityContextHolder.getContext();
    sc.setAuthentication(auth);
    HttpSession session = req.getSession(true);
    session.setAttribute(SPRING_SECURITY_CONTEXT_KEY, sc);
}
```

SPRING_SECURITY_CONTEXT_KEY is a statically imported *HttpSessionSecurityContextRepository.SPRING_SECURITY_CONTEXT_KEY*.

It should be noted that we can't directly use the *HttpSessionSecurityContextRepository* – because it works in conjunction with the *SecurityContextPersistenceFilter.*

That is because the filter uses the repository in order to load and store the security context before and after the execution of the rest of defined filters in the chain, but it uses a custom wrapper over the response which is passed to the chain.

So in this case, you should know the class type of the wrapper used and pass it to the appropriate save method in the repository.

4. Conclusion

(/)

In this quick tutorial, we went over how to manually set the user *Authentication* in the Spring Security context and how it can be made available for Spring MVC purposes, focusing on the code samples that illustrate the simplest way to achieve it.

As always, code samples can be found over on GitHub (https://github.com/eugenp/tutorials/tree/master/spring-security-modules/spring-security-web-mvc-custom).

COURSES

ALL COURSES (/COURSES/ALL-COURSES)

(/)

BAELDUNG ALL ACCESS (/COURSES/ALL-ACCESS)

BAELDUNG ALL TEAM ACCESS (/COURSES/ALL-ACCESS-TEAM)

THE COURSES PLATFORM (HTTPS://COURSES.BAELDUNG.COM)

SERIES

JAVA “BACK TO BASICS” TUTORIAL (/JAVA-TUTORIAL)

JACKSON JSON SERIES (/JACKSON)

APACHE HTTPCLIENT SERIES (/HTTPCLIENT-SERIES)

REST WITH SPRING SERIES (/REST-WITH-SPRING-SERIES)

SPRING PERSISTENCE SERIES (/PERSISTENCE-WITH-SPRING-SERIES)

SECURITY WITH SPRING (/SECURITY-SPRING)

SPRING REACTIVE SERIES (/SPRING-REACTIVE-SERIES)

ABOUT

ABOUT BAELDUNG (/ABOUT)

THE FULL ARCHIVE (/FULL_ARCHIVE)

EDITORS (/EDITORS)

OUR PARTNERS (/PARTNERS/)

PARTNER WITH BAELDUNG (/PARTNERS/WORK-WITH-US)

EBOOKS (/LIBRARY/)

FAQ (HTTPS://WWW.BAELDUNG.COM/LIBRARY/FAQ)

BAELDUNG PRO (/MEMBERS/)

TERMS OF SERVICE (/TERMS-OF-SERVICE)

PRIVACY POLICY (/PRIVACY-POLICY)

COMPANY INFO (/BAELDUNG-COMPANY-INFO)

CONTACT (/CONTACT)

(/)