# Custom Spring Boot Library for Binding Contextual Data

## The Problem

Our company decided to rewrite our existing platform using a modularized architecture. The new platform is made up of Dockerized Spring Boot apps deployed to Kubernetes. The modules can communicate with each other via REST and JMS messaging, both requiring JWT authentication. The JWT token contains reserved claims and custom claims that identify the user and provide contextual data (iss, sub, exp, permissions, etc). We wanted to authenticate the token for each REST and JMS message and make the contextual data available throughout the app.

### Our Solution

Our solution was simple. We decided to build a common library that would authenticate the JWT token for each REST and JMS message and then bind the contextual data to each thread via ThreadLocal to make the contextual data available throughout the app. This basically involved a Servlet filter, a WebClient, and a Spring aspect. An example of the common library can be found here: <https://github.com/valeriearena/contextdatalib-for-http-and-jms>.

##### The Common Library

Please NOTE: Some of the classes in the example have been renamed specifically for this article.

The common library (please see is used by all modules and is declared as a gradle dependency in each module. Please see each gradle build for moduleA and module.

The most important classes in the common library are the classes in the context package (ExampleContextData, ContextData, and ContextService), in addition to the CommonServletFilter, the CommonWebClient, and the CommonJmsInterceptor. Each class is described below.

The library also applies custom Spring Boot autoconfigurations that are required to add the JWT token to the REST message or the JMS message (in addition to customizing HttpSecurity so that each web request is authenticated).

The library does not have a Spring Boot launch class because the common library is not intended as a deployable app. Please see <https://github.com/valeriearena/contextdatalib-for-http-and-jms/tree/master/common>.

##### ExampleContextData, ContextData, and ContextService

ExampleContextData contains the specific context data for the user. It contains the actual JWT token, as well as the parsed claims contained in the JWT token.

ContextData is responsible for managing ExampleContextData and binds it to ThreadLocal and retrieves it from ThreadLocal.

ContextService manages ContextData. When ContextData needs to be added, ContextService adds it. When specific contextual data needs to be retrieved, ContextService retrieves it.

Please see <https://github.com/valeriearena/contextdatalib-for-http-and-jms/tree/master/common/src/main/java/com/example/common/context>.

##### CommonServletFilter

CommonServletFilter intercepts each web request to retrieve the JWT token from the Authorization header. CommonServletFilter authenticates the JWT token and parses it into ExampleContextData (via the JwtService). It then uses ContextService to bind ExampleContextData to ThreadLocal (via ContextData).

Please see <https://github.com/valeriearena/contextdatalib-for-http-and-jms/blob/master/common/src/main/java/com/example/common/filter/CommonFilter.java>.

##### CommonWebClient

The CommonWebClient retrieves the JWT token from ContextService and adds it to the Authorization header of the web request before submitting the request.

Please see https://github.com/valeriearena/contextdatalib-for-http-and-jms/tree/master/common/src/main/java/com/example/common/rest/webclient.

##### CommonJmsInterceptor

CommonJmsInterceptor intercepts each JMS message sent from JmsTopicPublisher and retrieves the JWT token from CotextService to add it to the Authorization property of the JMS message.

CommonJmsInterceptor also intercepts each JMS message received by JmsTopicSubscriber to retrieve the JWT token from the Authorization header. CommonJmsInterceptor authenticates the JWT token and parses it into ExampleContextData (via the JwtService). It then uses ContextService to bind ExampleContextData to ThreadLocal (via ContextData).

#### Github Example