**Validating JWT With Spring Boot and Spring Security**

For my current project I will have a REST API set up with Spring Boot. To be able to use the API endpoint the application will check that the incoming request has a valid JWT token.

[**[](https://dzone.com/users/382660/paskal_alma%40yahoo.com.html)**](https://dzone.com/users/382660/paskal_alma%40yahoo.com.html)**by**

[**Pascal Alma**](https://dzone.com/users/382660/paskal_alma%40yahoo.com.html)

**·**

**Jun. 15, 16 ·**[**Integration Zone**](https://dzone.com/enterprise-integration-training-tools-news)

**Like** [**(2)**](https://dzone.com/articles/validating-jwt-with-spring-boot-and-springsecurity)

**Comment (0)**

**Save**

[**Tweet**](https://dzone.com/articles/validating-jwt-with-spring-boot-and-springsecurity)

**5,052 Views**

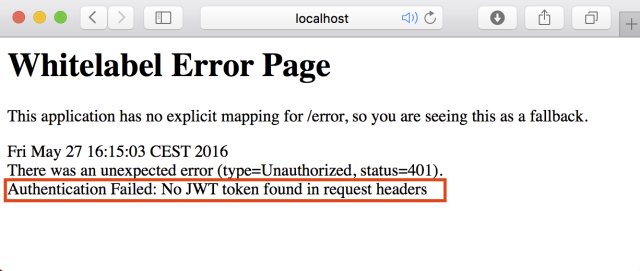
Join the DZone community and get the full member experience.

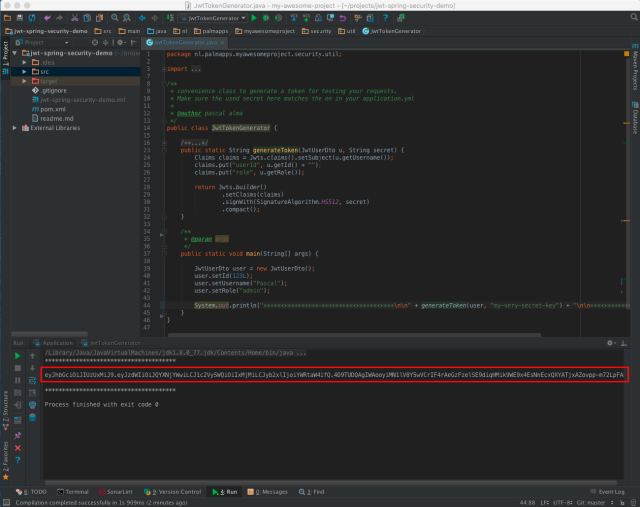
[**JOIN FOR FREE**](https://dzone.com/articles/validating-jwt-with-spring-boot-and-springsecurity)

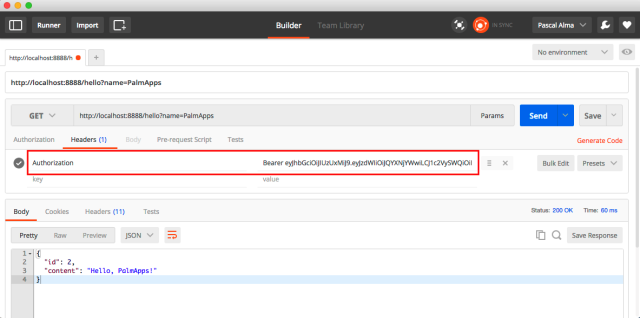
*Visually compose APIs with easy-to-use tooling. Learn how*[*IBM API Connect*](https://dzone.com/go?i=130034&u=http%3A%2F%2Fwww-03.ibm.com%2Fsoftware%2Fproducts%2Fen%2Fapi-connect)*provides near-universal access to data and services both on-premises and in the cloud, brought to you in partnership with*[*IBM*](https://dzone.com/go?i=130034&u=http%3A%2F%2Fwww-03.ibm.com%2Fsoftware%2Fproducts%2Fen%2Fapi-connect)*.*

For my current project I will have a [REST API](https://en.wikipedia.org/wiki/Representational_state_transfer) set up with [Spring Boot](http://projects.spring.io/spring-boot/) (most likely running with [BoxFuse](https://boxfuse.com/)). To be able to use the API endpoint the application will check that the incoming request has a valid [JWT token](https://jwt.io/) provided earlier (by an API service that I trust).

To implement this functionality I want to make use of [Spring Security](http://projects.spring.io/spring-security/) as it fits nicely with Spring Boot. When Googling for information about this combination I ran into [this site](https://www.toptal.com/java/rest-security-with-jwt-spring-security-and-java) that describes the background information quite nicely, but didn’t give me all the necessary sources to get it running. So after some more investigating and trial & error, I finally came to a working solution. Note that in my situation I only needed to validate an incoming token, I don’t need to create or supply new tokens.

The source code of the example can be found [here](https://gitlab.com/palmapps/jwt-spring-security-demo) on GitLab. The example application has a REST Controller called [MainController](https://gitlab.com/palmapps/jwt-spring-security-demo/blob/master/src/main/java/nl/palmapps/myawesomeproject/controller/MainController.java). After starting the application (by running the [Application](https://gitlab.com/palmapps/jwt-spring-security-demo/blob/master/src/main/java/nl/palmapps/myawesomeproject/controller/MainController.java).main method) you can access the REST endpoint with: <http://localhost:8888/hello?name=PalmApps>. As you will see you will get an HTTP 401 error if you try this in your browser:  
 [](https://pragmaticintegrator.wordpress.com/2016/05/27/validating-jwt-with-spring-boot-and-spring-security/screenshot-at-may-27-16-16-34/)

To get access to the endpoint you will need to supply a JWT token so you can get through the [JwtAuthenticationFilter](https://gitlab.com/palmapps/jwt-spring-security-demo/blob/master/src/main/java/nl/palmapps/myawesomeproject/security/JwtAuthenticationTokenFilter.java). To generate a valid token open the sources of the class [JwtTokenGenerator](https://gitlab.com/palmapps/jwt-spring-security-demo/blob/master/src/main/java/nl/palmapps/myawesomeproject/security/util/JwtTokenGenerator.java)and run the ‘main’ method, which will print a token in the console:  
[](https://pragmaticintegrator.wordpress.com/2016/05/27/validating-jwt-with-spring-boot-and-spring-security/screenshot-at-may-27-16-25-02/)

Copy the token and open a tool with which you can send an HTTP request and add the token to the header like [Postman](https://www.getpostman.com/):  
[](https://pragmaticintegrator.wordpress.com/2016/05/27/validating-jwt-with-spring-boot-and-spring-security/screenshot-at-may-27-16-28-09/) With the token in place you will see the expected output:

{

"id": 2,

"content": "Hello, PalmApps!"

}

If you access the endpoint <http://localhost:8888/me> with a POST request (still with the ‘Authorization’ header in place) you will get the details of the Principal object in JSON format:

{

"details": null,

"authorities": [

{

"authority": "admin"

}

],

"authenticated": true,

"principal": {

"username": "Pascal",

"token": "eyJhbGciOiJIUzUxMeJ9.eyJzdwIiOi....m72LpFADA",

"authorities": [

{

"authority": "admin"

}

],

"password": null

},

"credentials": null,

"name": "Pascal"

}

The ‘principal’ field in the returned object here is our [AuthenticatedUser](https://gitlab.com/palmapps/jwt-spring-security-demo/blob/master/src/main/java/nl/palmapps/myawesomeproject/security/model/AuthenticatedUser.java). If we get want to get more information from our JWT then we can simply add it to this object and fill it in the [JwtAuthenticationProvider](https://gitlab.com/palmapps/jwt-spring-security-demo/blob/master/src/main/java/nl/palmapps/myawesomeproject/security/JwtAuthenticationProvider.java).

*Visually compose APIs with easy-to-use tooling. Learn how*[*IBM API Connect*](https://dzone.com/go?i=130035&u=http%3A%2F%2Fwww-03.ibm.com%2Fsoftware%2Fproducts%2Fen%2Fapi-connect)*provides near-universal access to data and services both on-premises and in the cloud, brought to you in partnership with*[*IBM*](https://dzone.com/go?i=130035&u=http%3A%2F%2Fwww-03.ibm.com%2Fsoftware%2Fproducts%2Fen%2Fapi-connect)*.*