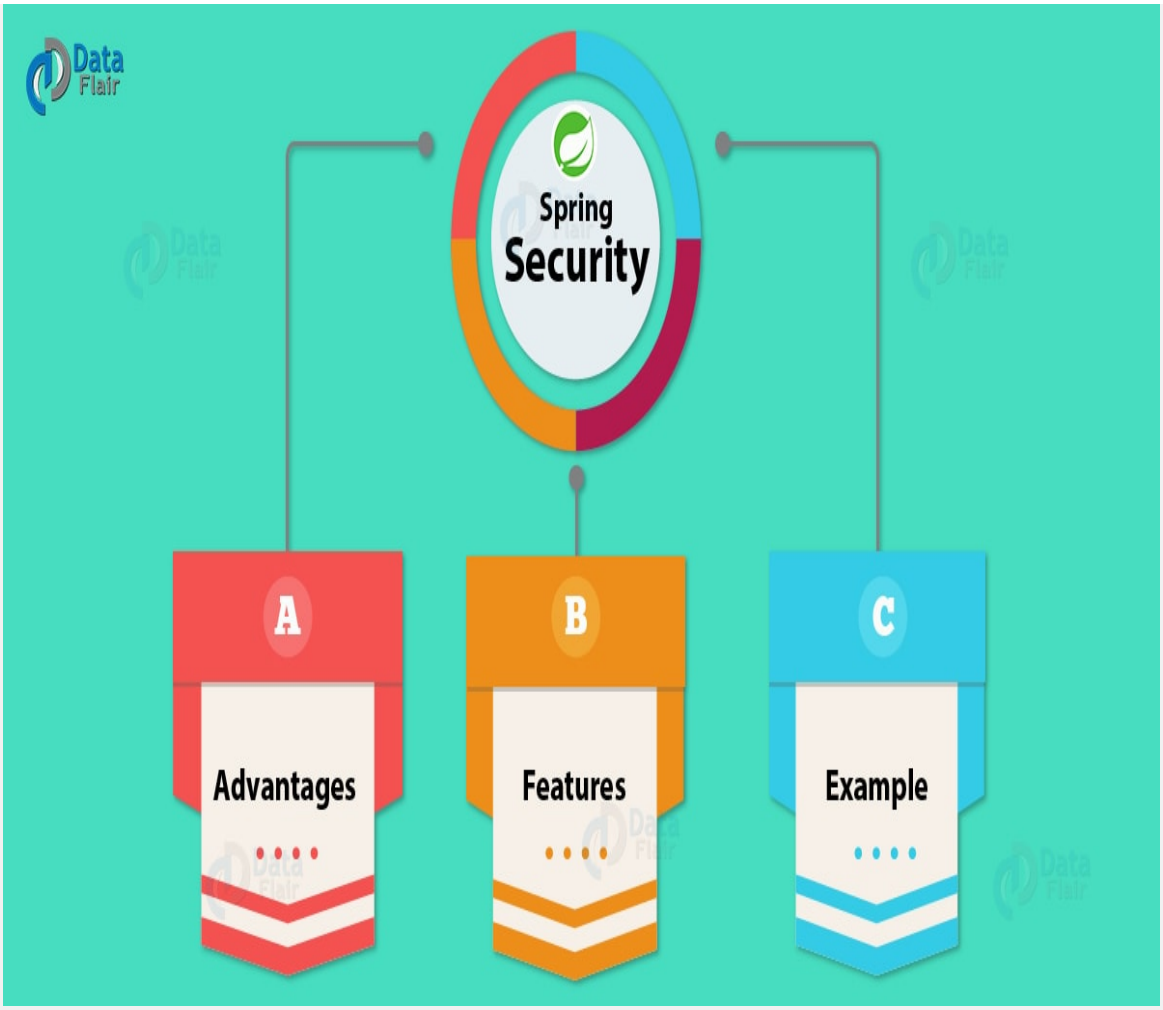
Spring Security Tutorial Step By Step – Example & Advantages

BY DATAFLAIR TEAM ·

1. Objective

In our last tutorial, we studied **Spring Web Services**. In this Spring Security tutorial, we are going to learn about security features of **Spring Framework**. Along with that, we will see the advantages of Spring Security and why security features are needed in Spring with an example using Eclipse IDE in place.
So, let's start the Spring Security Tutorial.



Spring Security Tutorial Step By Step – Example & Advantages

2. Spring Security

Before starting with the Spring security, one should have a basic knowledge of HTML and CSS. So that this article becomes understandable. Spring security is a framework that provides several security features. For example, authentication, authorization for creating secure Java Enterprise applications. This was a subproject which was started in 2003 by Ben Alex and later on in 2004, it was released as Spring Security 2.0.0 under the Apache license. It tries to overcome the problems that come from creating security applications with non-spring background and managing the new server environment.

Read about Spring Framework Environment Setup – How to Install Spring

Spring framework has two major applications which are authentication and authorization. Authentication is a process of identifying and knowing the target user which wants to access the application. Whereas Authorisation is a process of allowing the authority to perform some actions in the application. Also, you can apply authorization to a web request, methods and access to an individual domain.

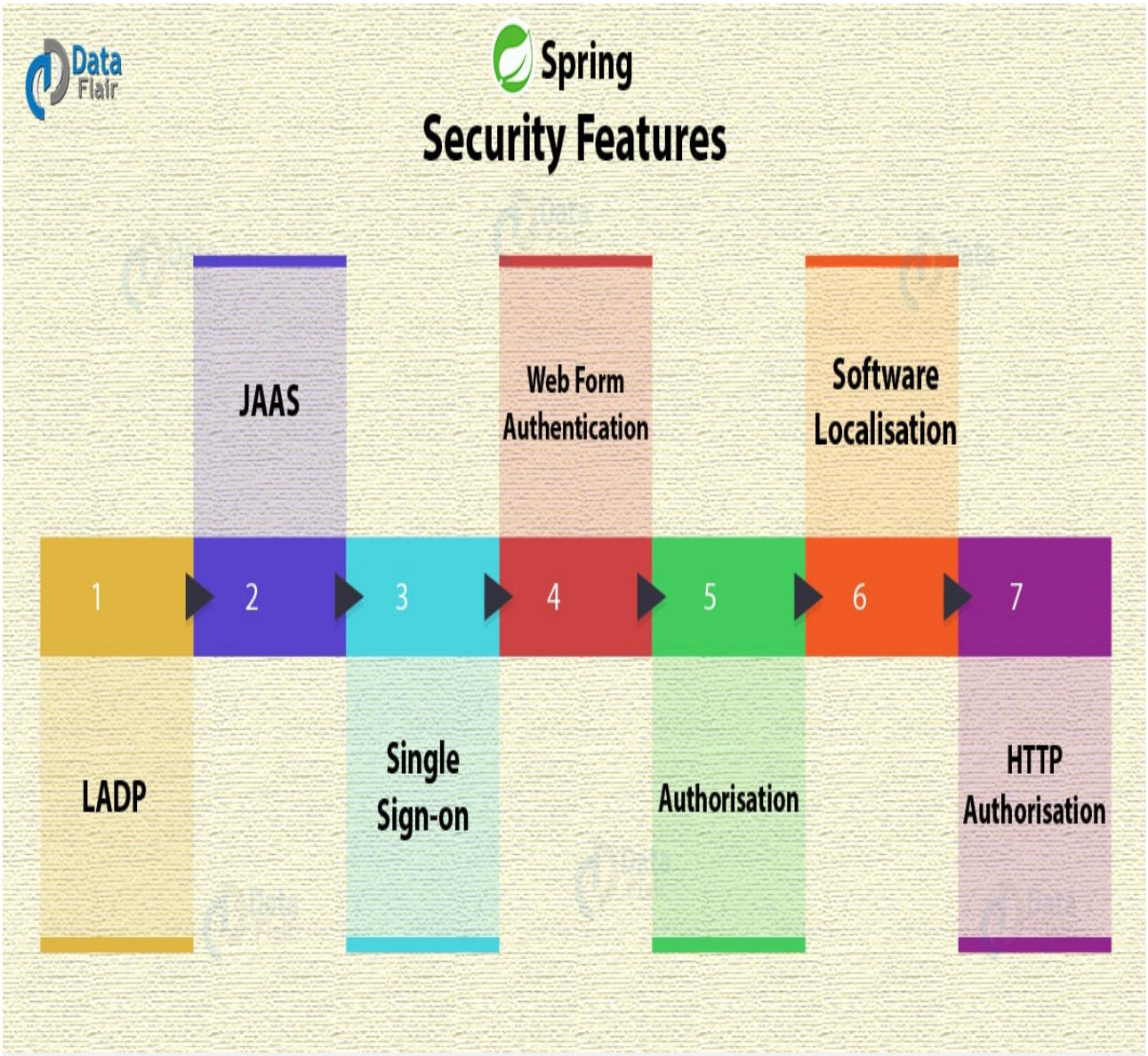
3. Advantages of Spring Security

The Spring Security Framework has several advantages. Some of them are listed:

- Configuration support to **Java Programming Language**.
- Portable.
- Comprehensive support to tasks like authorization and authentication.
- Servlet API integration.
- **Spring MVC integration**.
- CSRF protection.
- Protection against some common tasks.

4. Spring Security Features

Some of the Features of Spring Security Framework is as follows:



Features of Spring Security Framework

a. LADP

LADP stands for *Lightweight Directory Access Protocol*. It is an open application protocol which aims to maintain and access distributed directory information services using Internet Protocol (IP).

b. JAAS () Login Module

JAAS stands for *Java Authentication and Authorization Service*. This is a pluggable module which is implemented in Java and the Spring Security Framework uses it for its authentication purposes.

c. Single Sign-On

This allows the user to get access to multiple applications with only one account (username and password).

Let’s Explore Integration of Spring Logging with log4j – Eclipse IDE Coding

d. Web Form Authentication

In this the web form colleSpring Securitycts and authenticates user credentials from the web browser. The Spring Security supports so as to implement the web form authentication.

e. Authorisation

Its aim is to authorize the user giving it the access to the resources. This allows the developers to define the access policy against the valuable resources.

f. Software Localisation

It allows the developers to make the application UI in any language.

g. HTTP Authorisation

The Spring Framework provides this for HTTP authorization of the requests made by the web URLs using the Apache Ant paths or RE.

5. Spring Security Example

Now after getting to know about the Spring Security and its features, you will see a working example with Eclipse IDE in place. This example uses Java configuration and implements Spring Security without using XML.

Some of the steps are defined in order to make this project:

1. Create a Spring Security Java-based configuration which uses a Servlet Filter to protect application URLs.
2. Create the code for registering the springSecurityFilterChain for every URL in your application.
3. Loading the WebSecurityConfig in your existing ApplicationInitializer and add into the method getRootConfigClasses().
4. Use the class WebSecurityConfigurerAdapter to configure HttpSecurity http() containing default configuration.
5. Create a controller for handling user requests.
6. Run the above-created project and check for the valid credentials.

Read about Spring Transaction Management – Types and Methods

The code for WebSecurityConfig.java is as follows

```
package com.example;
import org.springframework.context.annotation.*;
import org.springframework.security.config.annotation.web.builders.HttpSecurity;
import org.springframework.security.config.annotation.web.configuration.*;
import org.springframework.security.core.userdetails.User;
import org.springframework.security.core.userdetails.UserDetailsService;
import org.springframework.security.provisioning.InMemoryUserDetailsManager;
import org.springframework.web.servlet.config.annotation.WebMvcConfigurer;
@EnableWebSecurity
@ComponentScan("com.example")
public class WebSecurityConfig implements WebMvcConfigurer {
@Bean
public UserDetailsService userDetailsService() throws Exception {
InMemoryUserDetailsManager manager = new InMemoryUserDetailsManager();
manager.createUser(User.withDefaultPasswordEncoder().username("example").
password("java123").roles("USER").build());
return manager;
}
protected void configure(HttpSecurity http) throws Exception {
http
.antMatcher("/")
.authorizeRequests()
.anyRequest().hasRole("ADMIN")
.and()
.httpBasic();
}
}
```

The code for SecurityWebApplicationInitializer.java is as defined

```
package com.example;
import org.springframework.security.web.context.*;
public class SecurityWebApplicationInitializer
extends AbstractSecurityWebApplicationInitializer {
}
```

The MvcWebApplicationInitializer.java is as follows

```
package com.example;
import
org.springframework.web.servlet.support.AbstractAnnotationConfigDispatcherServletIniti
alizer;
public class MvcWebApplicationInitializer extends
AbstractAnnotationConfigDispatcherServletInitializer {
@Override
protected Class<?>[] getRootConfigClasses() {
return new Class[] { WebSecurityConfig.class };
}
@Override
protected Class<?>[] getServletConfigClasses() {
// TODO Auto-generated method stub
return null;
}
@Override
protected String[] getServletMappings() {
return new String[] { "/" };
}
```

```
}
```

The default configuration for WebSecurityConfigurerAdapeter class is as defined

```
protected void configure(HttpSecurity http) throws Exception {
http
.authorizeRequests()
.anyRequest().authenticated()
.and()
.formLogin()
.and()
.httpBasic();
}
```

Do you know about Spring Bean Definition

The controller Java file named HomeController.java is as defined

```
package com.example.controller;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
@Controller
public class HomeController {
@RequestMapping(value="/", method=RequestMethod.GET)
public String index() {
return "index";
}
} <span style="background-color: #fafafa;color: #333333;font-family: Verdana, Geneva, sans-serif;font-size: 16px;font-weight: inherit"> </span>
```

Creating a view page index.jsp having the following source code

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Home Page</title>
</head>
<body>
Welcome to home page!
</body>
</html>
```

Once you are done creating the Java files run the above-created project. If you try logging with the wrong credentials it will prompt you the following message as output:

Your login attempt was not successful, try again.

Reason: Bad credentials

If you then try logging with correct credentials you will be directed to another page having the following message:

Welcome to home page!

6. Conclusion

Hence, in this Spring Security tutorial, we studied the Spring Security Framework. Along with that, you saw its features, advantages and a working example using Eclipse IDE. Going with the session you have got the basic knowledge needed for creating a user login page having features such as authentication and authorization. Still, if you have any query, feel free to ask in the comment section.
See Also- **Spring Bean Scope**
For reference