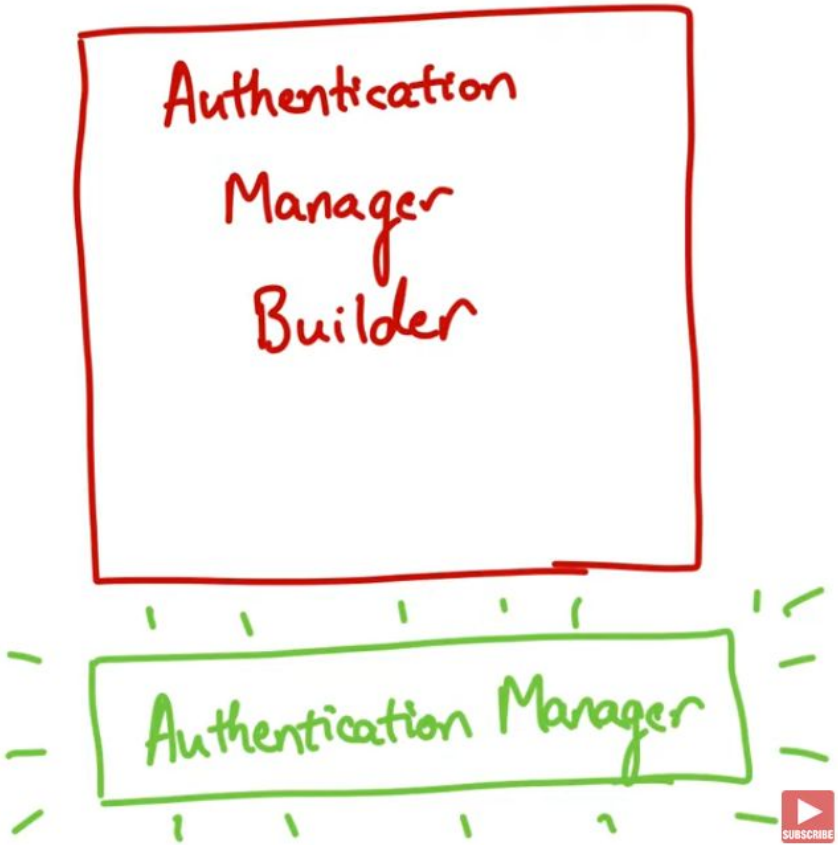
## Starting point

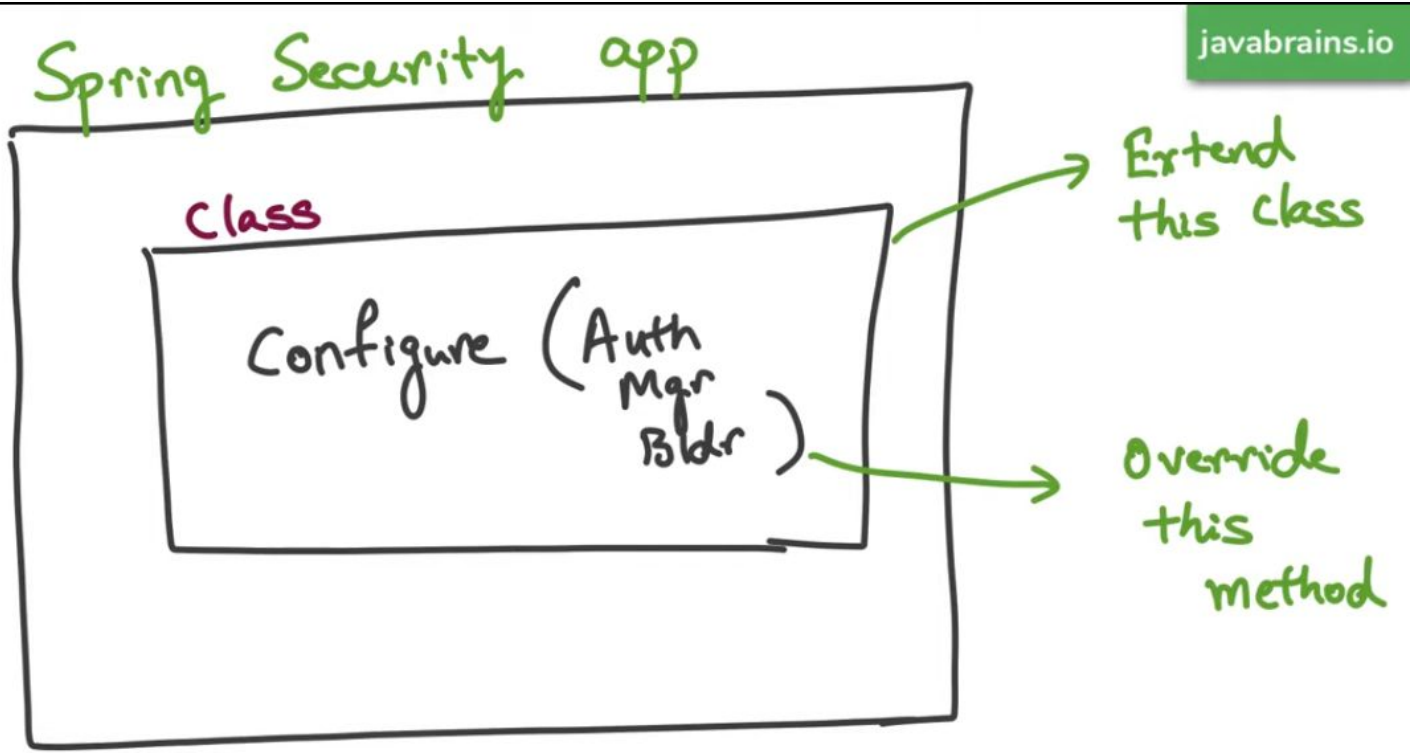
# A simple Spring Boot application with Spring Security dependency added

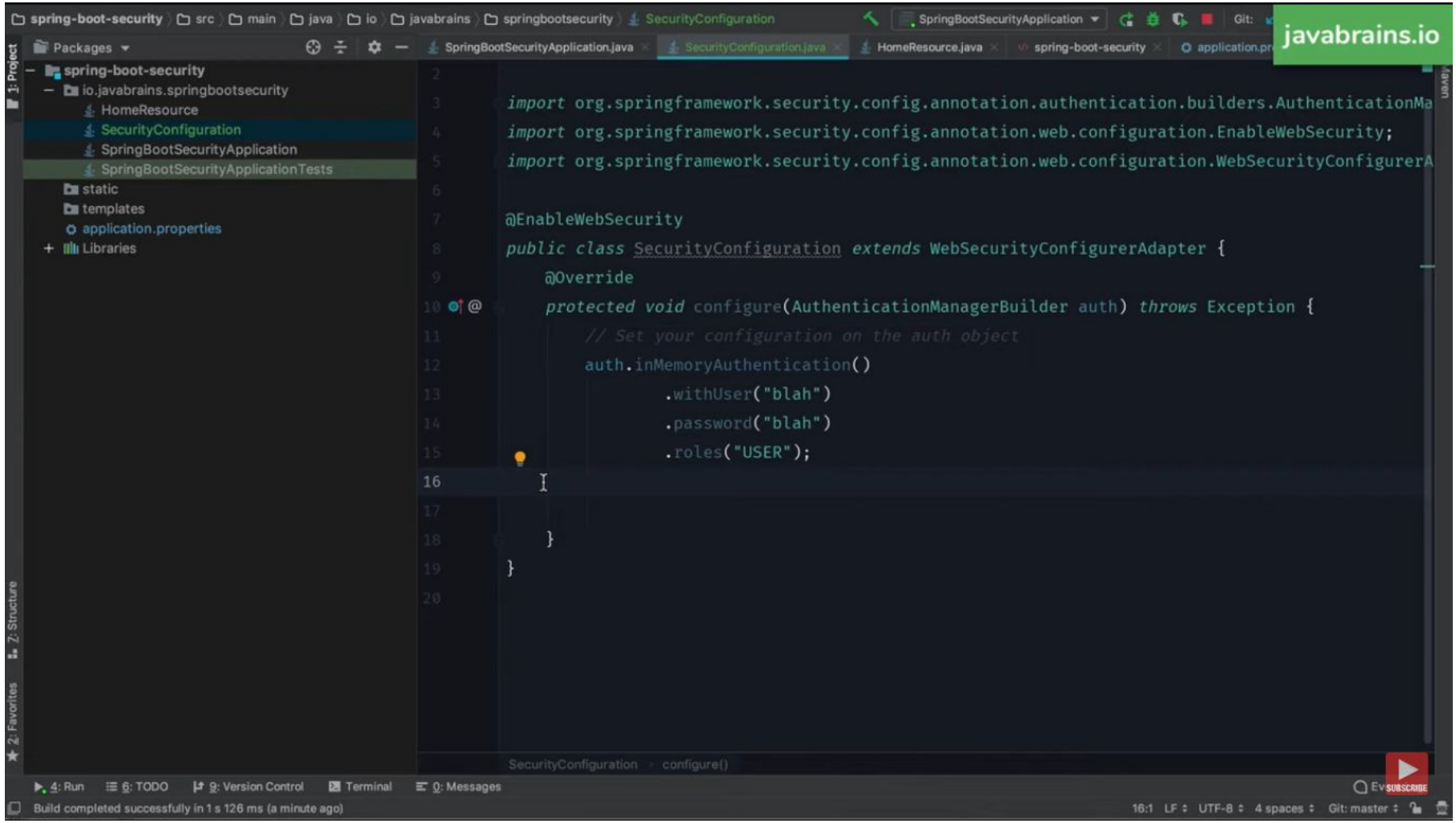
# Steps

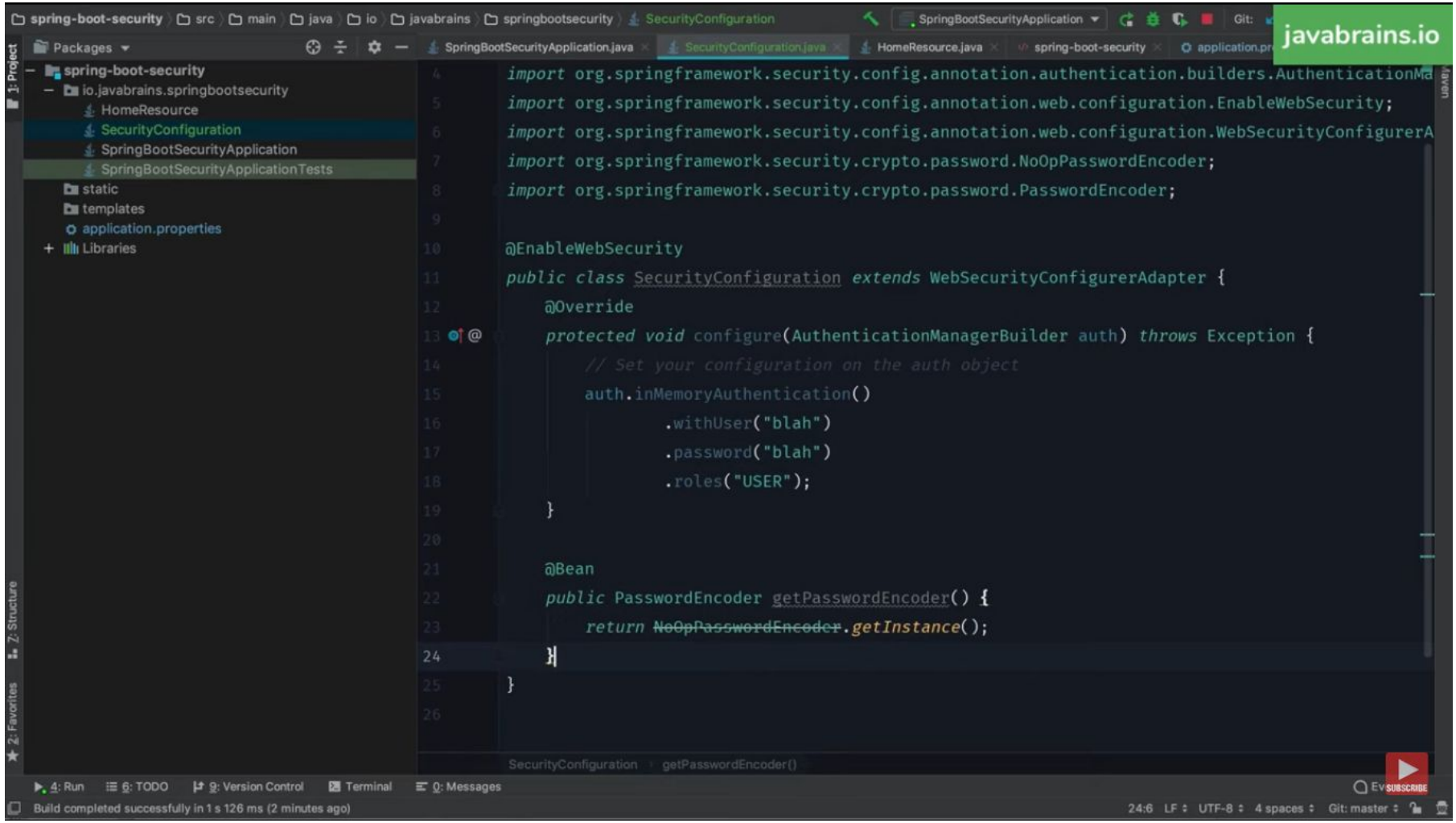
**Step 1:** Get hold of AuthenticationManagerBuilder

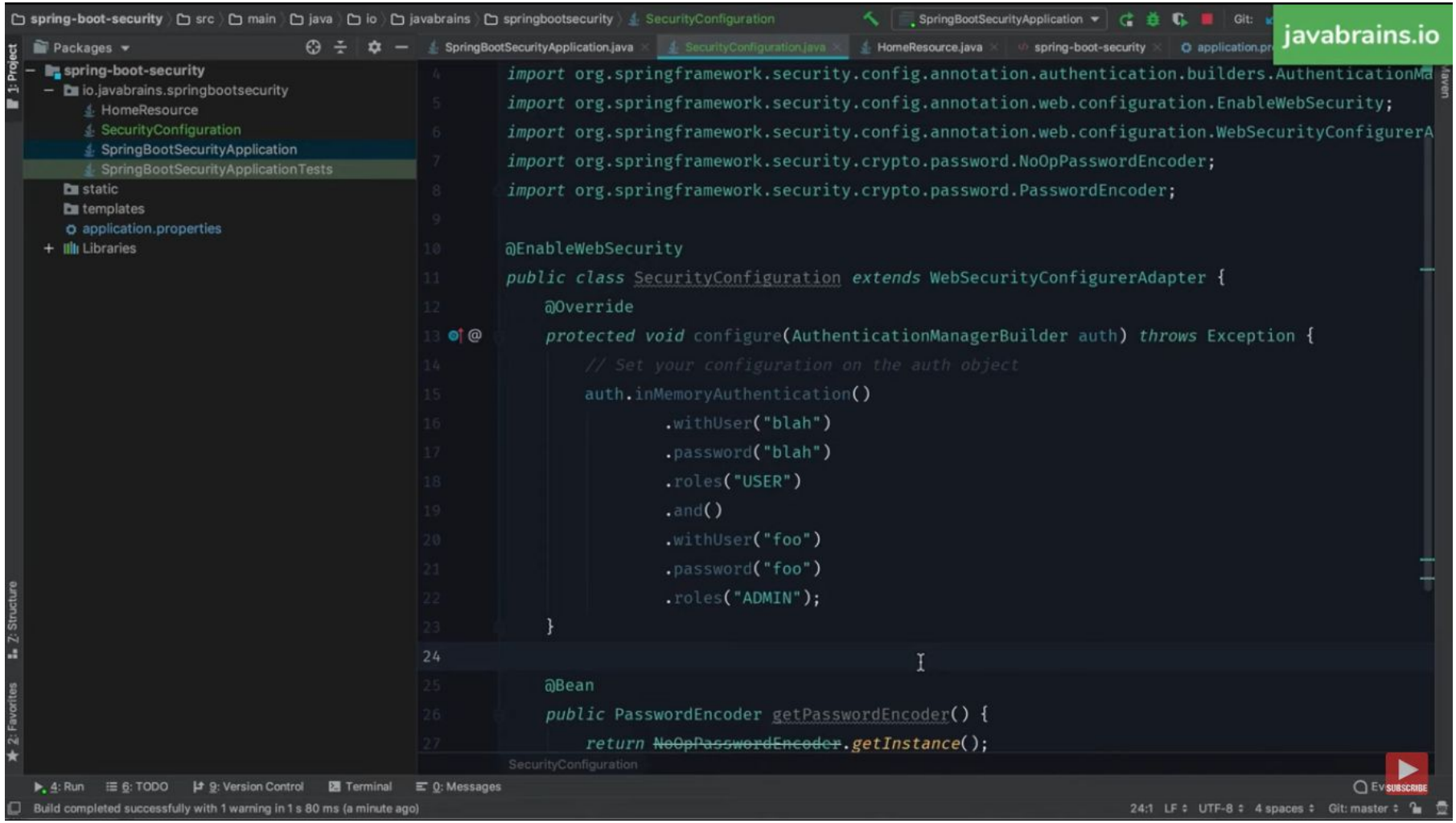
**Step 2:** Set the configuration on it

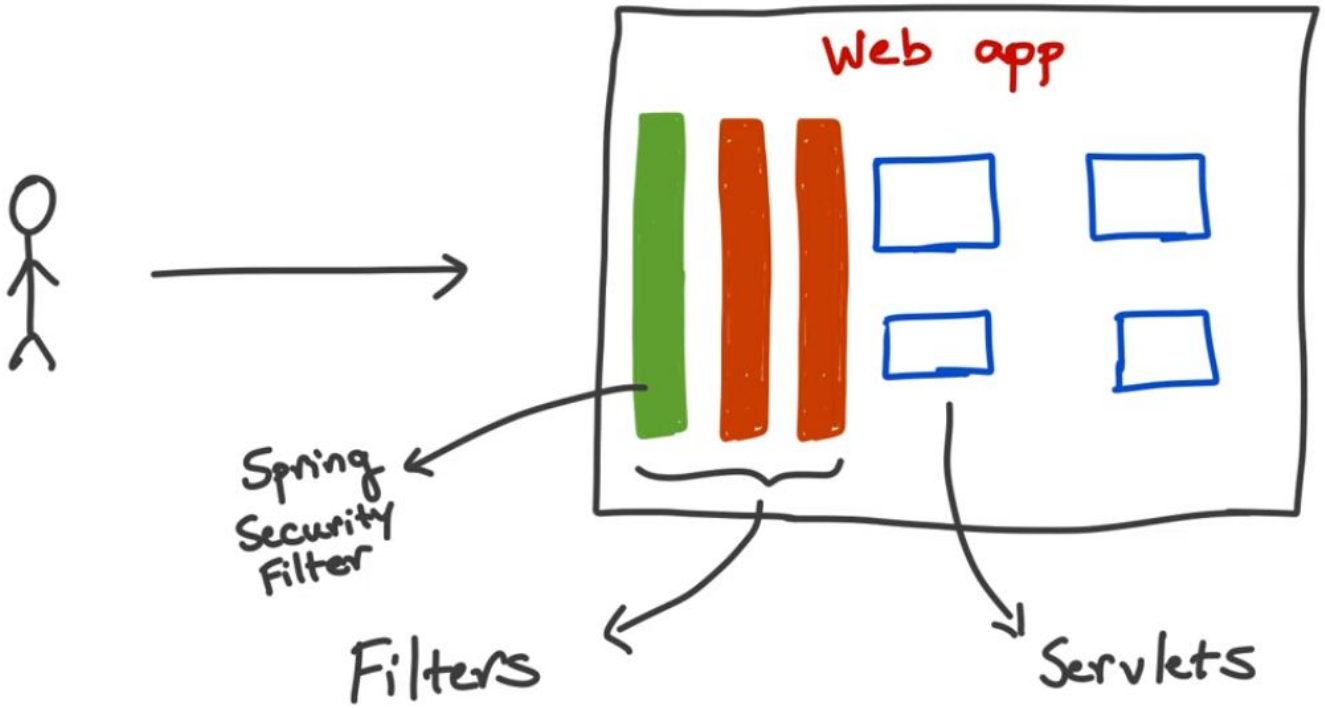

Authentication
Manager
Builder
Authentication Manager
SUBSCRIBE


javabrains.io
Spring Security app
Class
Configure (Auth Mgr Bldr)
Extend this class
Override this method


spring-boot-security
src
main
java
io
javabrains
springbootsecurity
SecurityConfiguration
SpringBootSecurityApplication
Git:
javabrains.io
Packages
SpringBootSecurityApplication.java
SecurityConfiguration.java
HomeResource.java
spring-boot-security
application.pr
spring-boot-security
io.javabrains.springbootsecurity
HomeResource
SecurityConfiguration
SpringBootSecurityApplication
SpringBootSecurityApplicationTests
static
templates
application.properties
Libraries
import org.springframework.security.config.annotation.authentication.builders.AuthenticationMa
import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
import org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerA
@EnableWebSecurity
public class SecurityConfiguration extends WebSecurityConfigurerAdapter {
@Override
protected void configure(AuthenticationManagerBuilder auth) throws Exception {
// Set your configuration on the auth object
auth.inMemoryAuthentication()
.withUser("blah")
.password("blah")
.roles("USER");
}
}
SecurityConfiguration
configure()
4: Run
6: TODO
9: Version Control
Terminal
0: Messages
Build completed successfully in 1 s 126 ms (a minute ago)
16:1
LF
UTF-8
4 spaces
Git: master
SUBSCRIBE


spring-boot-security
src
main
java
io
javabrains
springbootsecurity
SecurityConfiguration
SpringBootSecurityApplication
Git:
javabrains.io
Packages
SpringBootSecurityApplication.java
SecurityConfiguration.java
HomeResource.java
spring-boot-security
application.pr
spring-boot-security
io.javabrains.springbootsecurity
HomeResource
SecurityConfiguration
SpringBootSecurityApplication
SpringBootSecurityApplicationTests
static
templates
application.properties
Libraries
import org.springframework.security.config.annotation.authentication.builders.AuthenticationMa
import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
import org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerA
import org.springframework.security.crypto.password.NoOpPasswordEncoder;
import org.springframework.security.crypto.password.PasswordEncoder;
@EnableWebSecurity
public class SecurityConfiguration extends WebSecurityConfigurerAdapter {
@Override
protected void configure(AuthenticationManagerBuilder auth) throws Exception {
// Set your configuration on the auth object
auth.inMemoryAuthentication()
.withUser("blah")
.password("blah")
.roles("USER");
}
@Bean
public PasswordEncoder getPasswordEncoder() {
return NoOpPasswordEncoder.getInstance();
}
}
SecurityConfiguration
getPasswordEncoder()
SUBSCRIBE
4: Run
6: TODO
9: Version Control
Terminal
0: Messages
Build completed successfully in 1 s 126 ms (2 minutes ago)
24:6
LF
UTF-8
4 spaces
Git: master

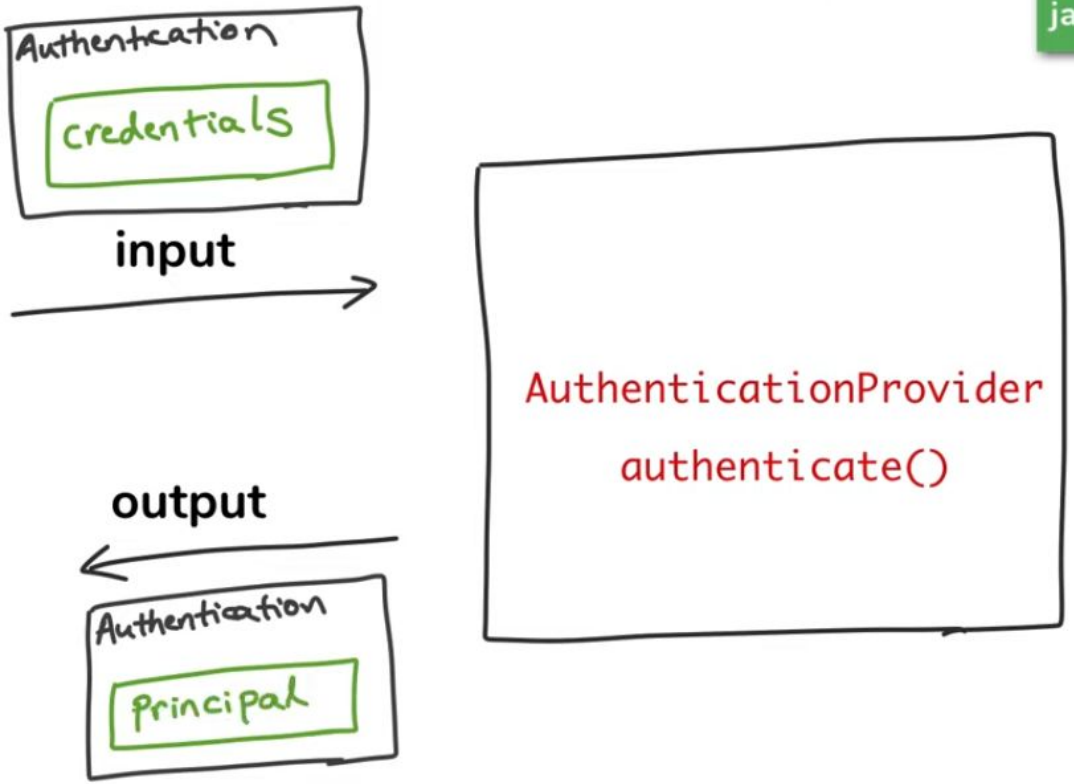

spring-boot-security
src
main
java
io
javabrains
springbootsecurity
SecurityConfiguration
SpringBootSecurityApplication
Git:
javabrains.io
Packages
SpringBootSecurityApplication.java
SecurityConfiguration.java
HomeResource.java
spring-boot-security
application.pr
spring-boot-security
io.javabrains.springbootsecurity
HomeResource
SecurityConfiguration
SpringBootSecurityApplication
SpringBootSecurityApplicationTests
static
templates
application.properties
Libraries
import org.springframework.security.config.annotation.authentication.builders.AuthenticationMa
import org.springframework.security.config.annotation.web.configuration.EnableWebSecurity;
import org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerA
import org.springframework.security.crypto.password.NoOpPasswordEncoder;
import org.springframework.security.crypto.password.PasswordEncoder;
@EnableWebSecurity
public class SecurityConfiguration extends WebSecurityConfigurerAdapter {
@Override
protected void configure(AuthenticationManagerBuilder auth) throws Exception {
// Set your configuration on the auth object
auth.inMemoryAuthentication()
.withUser("blah")
.password("blah")
.roles("USER")
.and()
.withUser("foo")
.password("foo")
.roles("ADMIN");
}
@Bean
public PasswordEncoder getPasswordEncoder() {
return NoOpPasswordEncoder.getInstance();
SecurityConfiguration
4: Run
6: TODO
9: Version Control
Terminal
0: Messages
Build completed successfully with 1 warning in 1 s 80 ms (a minute ago)
24:1
LF
UTF-8
4 spaces
Git: master
SUBSCRIBE



javabrains.io

bootstrap | authentication

# Providers



org.springframework.security.authentication

## Interface AuthenticationProvider

---

`public interface` **`AuthenticationProvider`**

Indicates a class can process a specific `Authentication` implementation.

### *Method Summary*

All Methods | Instance Methods | Abstract Methods

| Modifier and Type | Method and Description |
| --- | --- |
| `Authentication` | **`authenticate(Authentication`** `authentication)`<br>Performs authentication with the same contract as **`AuthenticationManager.authenticate(Authentication)`**. |

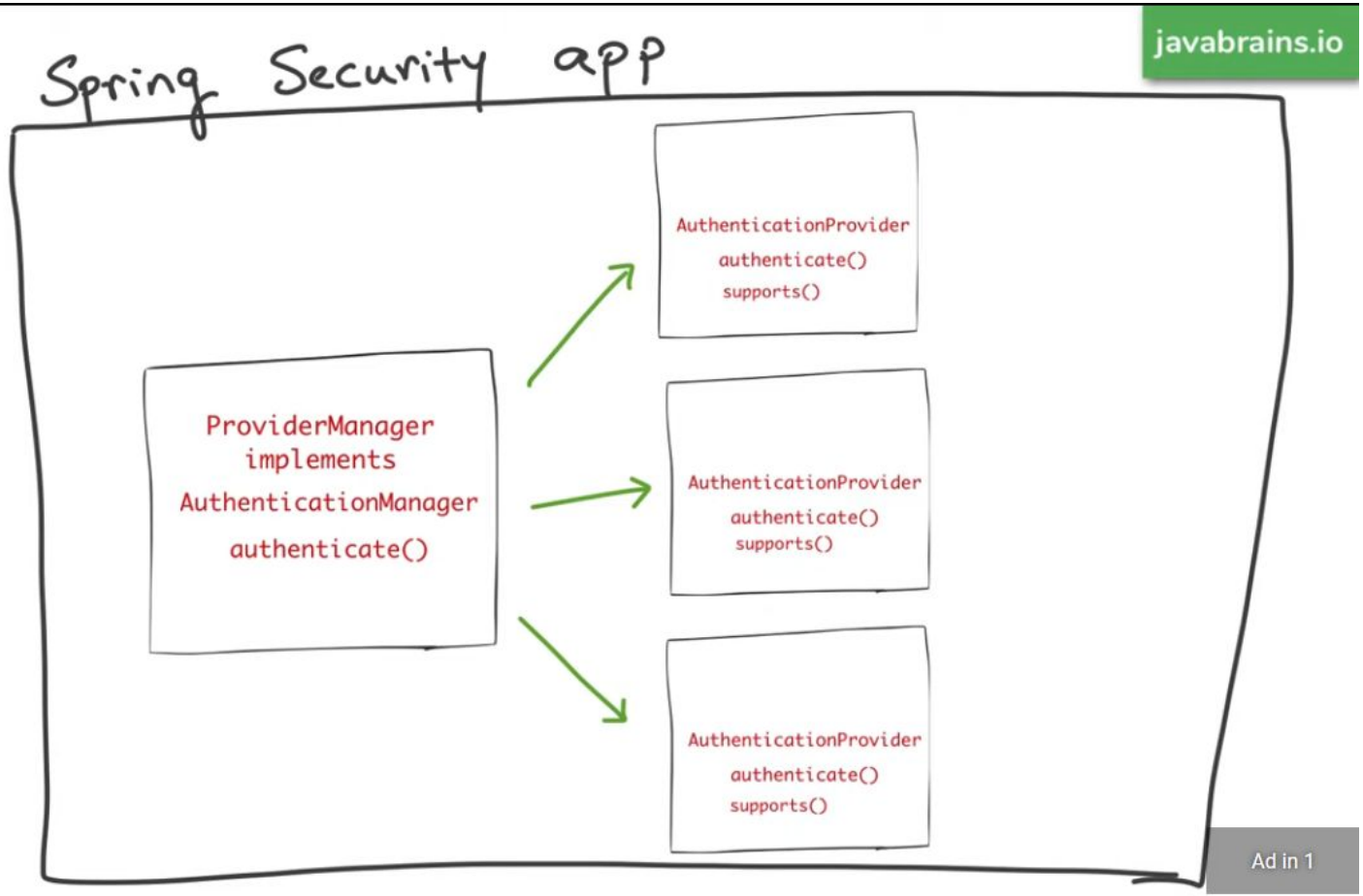
org.springframework.security.core

## Interface Authentication

### *Method Summary*

All Methods | Instance Methods | Abstract Methods

| Modifier and Type | Method and Description |
|---|---|
| `java.util.Collection<? extends GrantedAuthority>` | **`getAuthorities()`**<br>Set by an `AuthenticationManager` to indicate the authorities that the principal has been granted. |
| `java.lang.Object` | **`getCredentials()`**<br>The credentials that prove the principal is correct. |
| `java.lang.Object` | **`getDetails()`**<br>Stores additional details about the authentication request. |
| `java.lang.Object` | **`getPrincipal()`**<br>The identity of the principal being authenticated. |
| `boolean` | **`isAuthenticated()`**<br>Used to indicate to `AbstractSecurityInterceptor` whether it should present the authentication token to the `AuthenticationManager`. |
| `void` | **`setAuthenticated(boolean isAuthenticated)`**<br>See **`isAuthenticated()`** for a full description. |


Spring Security app
javabrains.io
ProviderManager
implements
AuthenticationManager
authenticate()
AuthenticationProvider
authenticate()
supports()
AuthenticationProvider
authenticate()
supports()
AuthenticationProvider
authenticate()
supports()

org.springframework.security.authentication

## Interface AuthenticationProvider

---

`public interface` **`AuthenticationProvider`**

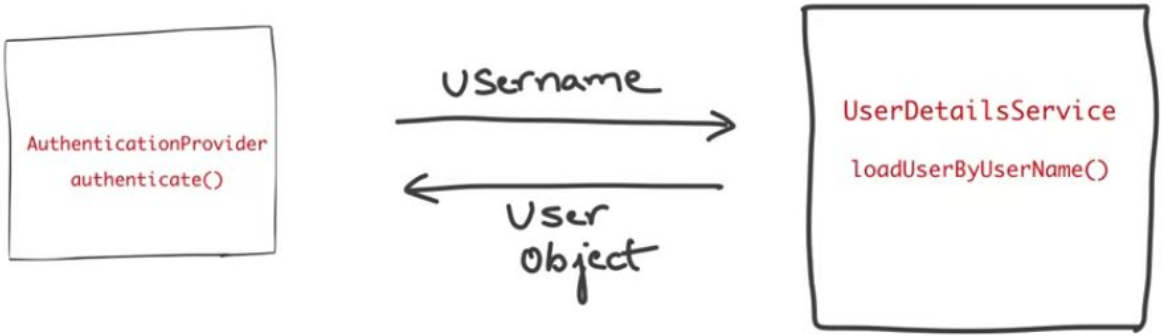
Indicates a class can process a specific `Authentication` implementation.

### *Method Summary*

All Methods | Instance Methods | Abstract Methods

| Modifier and Type | Method and Description |
|---|---|
| `Authentication` | **`authenticate`**`(`**`Authentication`**` authentication)`<br>Performs authentication with the same contract as **`AuthenticationManager.authenticate(Authentication)`**. |
| `boolean` | **`supports`**`(java.lang.Class<?> authentication)`<br>Returns `true` if this `AuthenticationProvider` supports the indicated `Authentication` object. |


AuthenticationProvider
authenticate()
Username
User
Object
Identity
Store


AuthenticationProvider
authenticate()
Username
User
Object
UserDetailsService
loadUserByUserName()

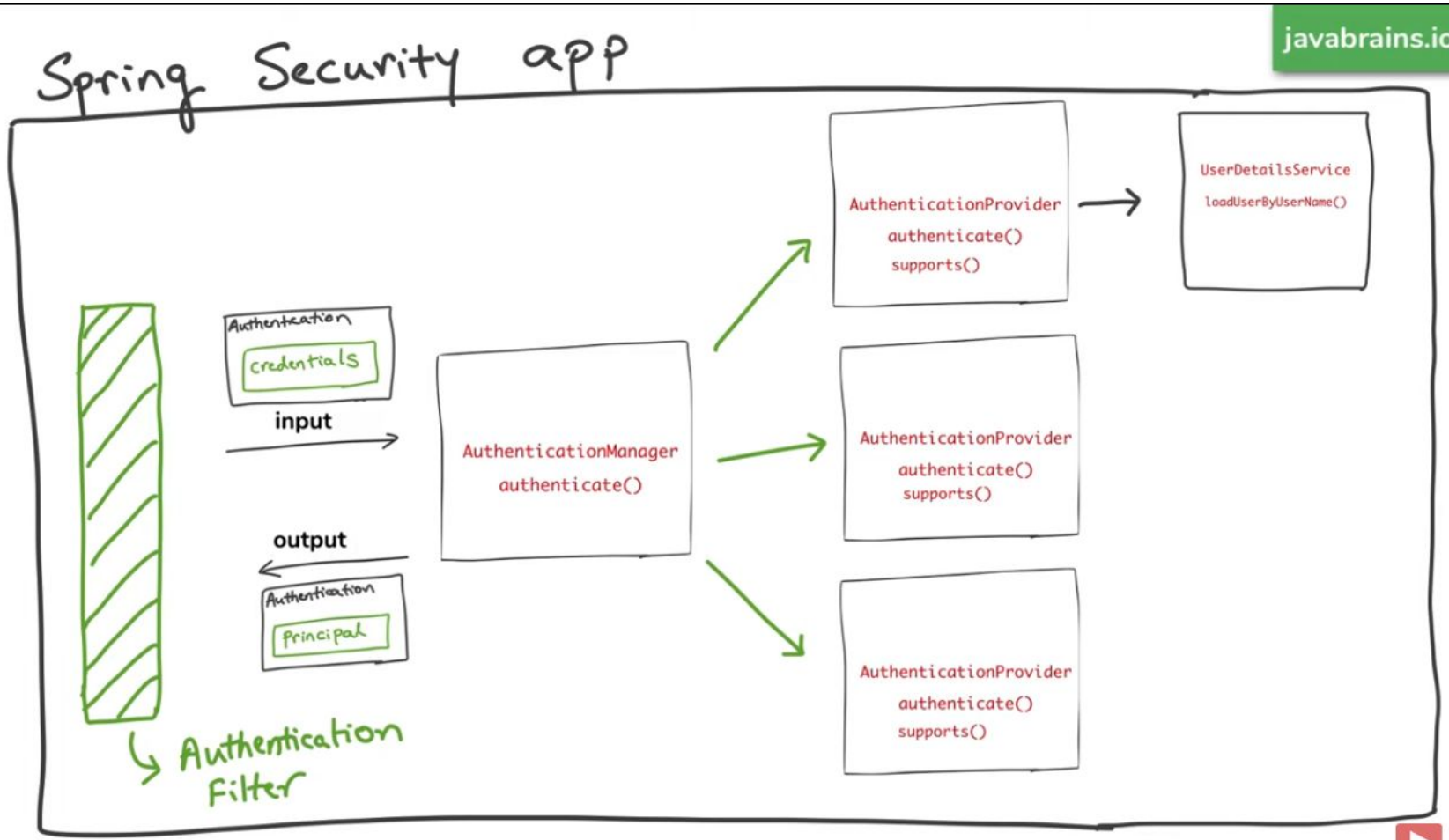
org.springframework.security.core.userdetails

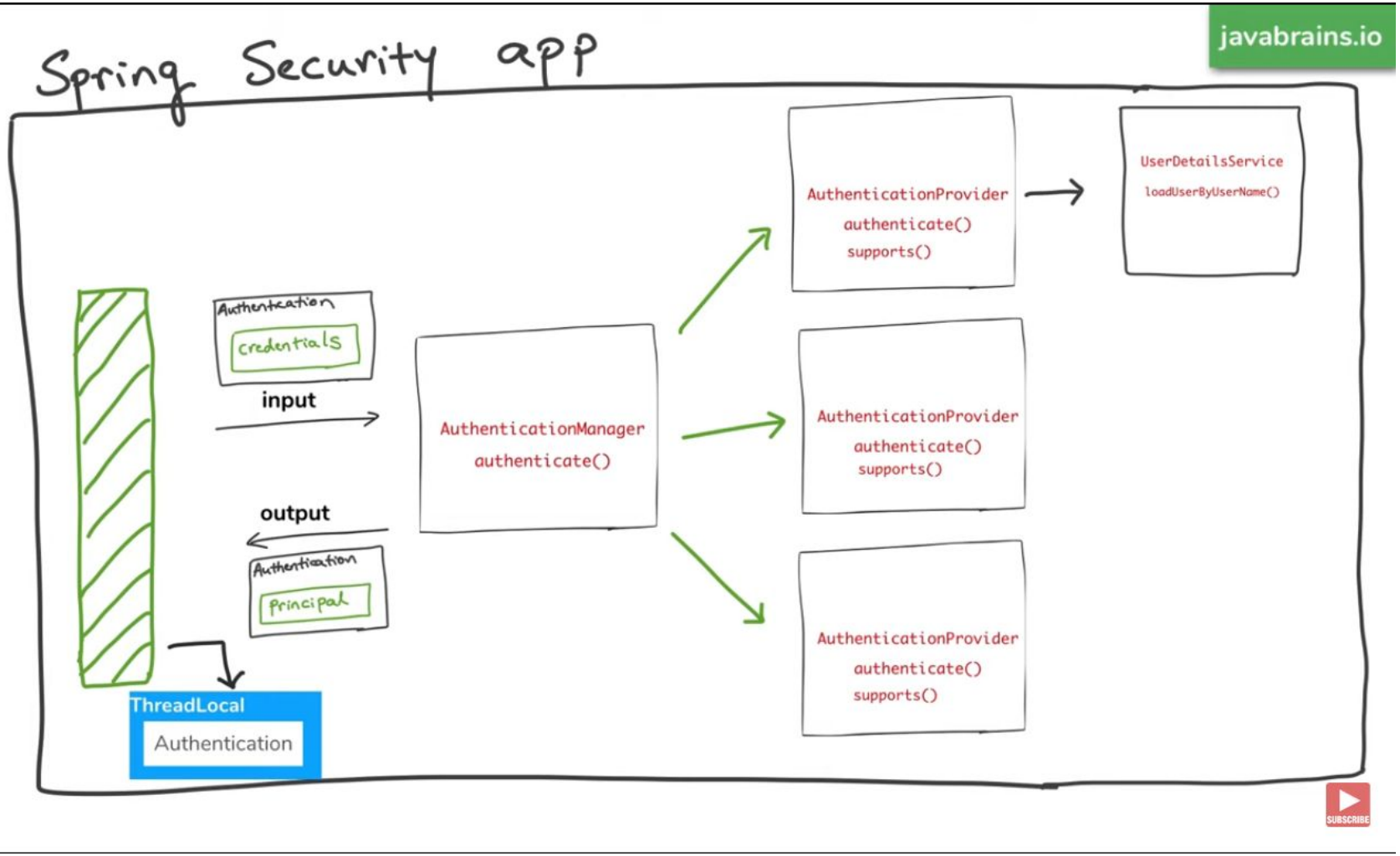
## Interface UserDetails

### *Method Summary*

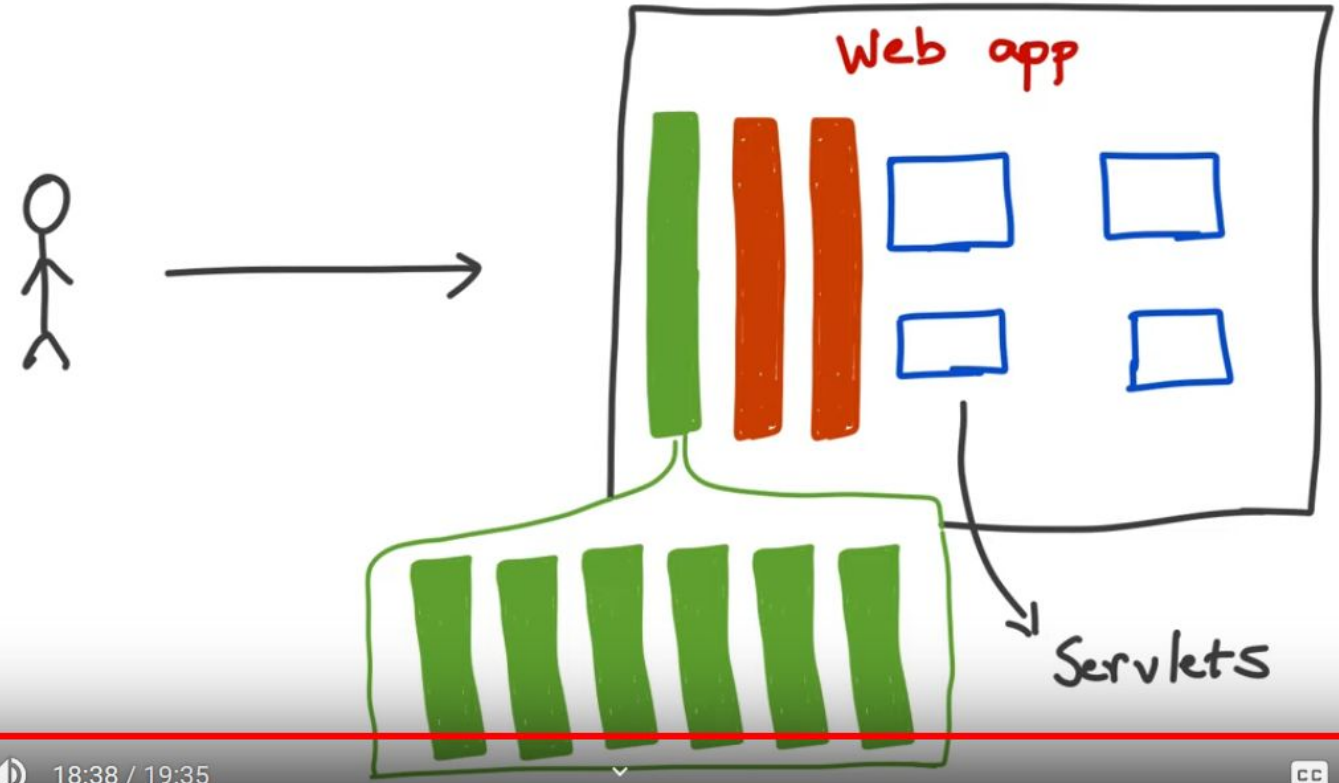
All Methods | Instance Methods | Abstract Methods

| Modifier and Type | Method and Description |
|---|---|
| `java.util.Collection<? extends GrantedAuthority>` | **`getAuthorities()`**<br>Returns the authorities granted to the user. |
| `java.lang.String` | **`getPassword()`**<br>Returns the password used to authenticate the user. |
| `java.lang.String` | **`getUsername()`**<br>Returns the username used to authenticate the user. |
| `boolean` | **`isAccountNonExpired()`**<br>Indicates whether the user's account has expired. |
| `boolean` | **`isAccountNonLocked()`**<br>Indicates whether the user is locked or unlocked. |
| `boolean` | **`isCredentialsNonExpired()`**<br>Indicates whether the user's credentials (password) has expired. |
| `boolean` | **`isEnabled()`**<br>Indicates whether the user is enabled or disabled. |


Spring Security app
javabrains.io
AuthenticationProvider
authenticate()
supports()
UserDetailsService
loadUserByUserName()
Authentication
credentials
input
AuthenticationManager
authenticate()
AuthenticationProvider
authenticate()
supports()
output
Authentication
Principal
AuthenticationProvider
authenticate()
supports()
Authentication Filter
SUBSCRIBE


Spring Security app
javabrains.io
AuthenticationProvider
authenticate()
supports()
UserDetailsService
loadUserByUserName()
Authentication
credentials
input
AuthenticationManager
authenticate()
AuthenticationProvider
authenticate()
supports()
output
Authentication
Principal
AuthenticationProvider
authenticate()
supports()
ThreadLocal
Authentication
SUBSCRIBE

How Spring Security Authentication works - Java Brains




18:38 / 19:35