技術筆記

筆記更新紀錄

|  |  |  |
| --- | --- | --- |
| 日期 yyyy/MM/dd | 更新人員 | 更新備註 |
| 2020/02/15 | Miles | 建立文件 |
| 2020/02/18 | Miles | 添加在不使用任何配置的情況之下預設的Security |
| 2020/02/19 | Miles | 添加不需要放置@EnableWebSecurity的原因 |
|  |  |  |
|  |  |  |
|  |  |  |

目錄

[1關於Spring Security 4](#_Toc33027266)

[2剖析Spring Security 概念 4](#_Toc33027267)

[3 不需要使用EnableWebSecurity 的原因 4](#_Toc33027268)

[3使用Spring Security(預設情況) 6](#_Toc33027269)

[3使用Spring Security(UserDetails 與 User) 8](#_Toc33027270)

[4使用Spring Security(身分驗證管理器) 10](#_Toc33027271)

[3使用Spring Security(身分驗證管理器-自定義使用者權限) 14](#_Toc33027272)

[99其他語法與效果 15](#_Toc33027273)

## 1關於Spring Security

|  |
| --- |
| Spring Security 是什麼? |
| Spring Security是一個基於Java / Java EE的框架，為企業應用程式提供身份驗證，授權和其他安全功能。為Spring的其中一個模組。 |

## 2剖析Spring Security 概念

|  |
| --- |
| 1 安全認證其主要分成兩個部分(Spring Security 核心主要功能) |
| 1 認證(Authentication) |
| 2 授權(Authorization) |

|  |
| --- |
| 2 其實就是一連串的Filter |
| 實現了各種Filter 其原理使用AOP的改念。 |

## 3 不需要使用EnableWebSecurity 的原因

|  |
| --- |
| 3 Spring Boot 幫我啟動了Spring Security (核心概念) |
| 當我們使用了Spring Boot 或者是 Spring Cloud 的starter-config時，Spring 的factories會自動的呼叫EnableAutoConfiguration 並且指向SecurityAUtoConfiguration 類別。    該類別會Import三個類別    1 SpringBootWebSecurityConfiguration.class  2 WebSecurityEnablerConfiguration.class  3 SecurityDataConfiguration.class  如果有導入Spring Security 相關jar檔(core) 那麼該類別(SecurityAutoConfiguration將會自動啟用。[其中WebSecurityEnablerConfiguration.class當中有一個 @EnableWebSecurity](mailto:其中WebSecurityEnablerConfiguration.class當中有一個@EnableWebSecurity)會幫我們啟動Spring Security 功能。 |
| 結論就是 當我們使用Spring Boot 或者是Spring Cloud 的時候，並不需要取啟用@EnableWebSecurity 這個Annotation。 |
| 如果無法類似以上的方式找尋Auto Configurar 找到該Annotation，那就安全起見，還是要加上@EnableWebSecurity這個Annotation。 |

## 4關於EnableWebSecurity 的運作

|  |
| --- |
| 進入原始碼看說明 |
| 1. 添加這個Annotation在@Configuration 的類別上就能擁有Spring Security功能。 |
|  |

## 3使用Spring Security(預設情況)

|  |
| --- |
| 1 導入Lib |
| 因為使用微服務，所以可以直接在Gradle導入    範例暫時使用2.1.3.RELEASE 版本。  如果非使用微服務，可導入以下Lib |

|  |
| --- |
| 2 寫一個測試的RestController |
| 我們寫了一個入口 |

|  |
| --- |
| 3 啟動這個專案 |
| 因為我們沒有做任何Security的配置，所以這時候Spring Security的容器會幫我們產生一個預設的密碼    以上Using generated security password 就是預設的密碼。帳號則是user |

|  |
| --- |
| 4 瀏覽器測試結果 |
| 瀏覽器輸入localhost:XXXX/test 會出現以下畫面:    特別注意，該畫面是Spring提供的httpBasicLogin畫面，做為測試用。  輸入帳號user 與密碼:#######    如果我們把URL輸入改成localhost:XXXX/logout    以上詢問是否登出的畫面也是Spring提供的測試用畫面 |

## 3使用Spring Security(UserDetails 與 User)

|  |
| --- |
| 1 UserDetails 是什麼? |
| 這是一個提供提取使用者資訊的介面。 |

|  |
| --- |
| 2 User 是什麼? |
| 這是一個存放使用者的資訊。該類別實作了UserDetails介面，所以該物件具備提取使用者資訊的方法。 |

|  |
| --- |
| 2.1 User 有哪些參數 |
| 我們進入原始碼可以看到    1 password : 使用者密碼  2 username : 使用者名稱  3 authorities :　使用權限  4 accountNonExpired : 帳戶是否過期  5 accountNonLocked : 帳戶是否凍結  6 credentialsNonExpired : 使用者密碼是否過期  7 enabled : 帳號是否可使用 |

|  |
| --- |
| 3 建立一個UserDetails ，他需要UserDetailsService 幫忙產生! |
| 因為SpringSecurity 會自動呼叫UserDetailsService介面(前提是你要啟用Spring Security)  所以當我們要使用時，可以直接Overrid 這個介面，讓Spring Security呼叫我們也得UserDetailsService    該介面只有一個方法需要實作    時候做後可以使用    該Service便會產生UserDetails的物件 |

|  |
| --- |
| 4 建立User |
| 以上流程會幫我們產生UserDetails 進行使用者的比對，這時候我們要產生User物件進行存放。 |

|  |
| --- |
| 5 整個建立的流程大致上這樣… |
| 當我們進入檢 |

## 4使用Spring Security(身分驗證管理器)

|  |
| --- |
| 1 導入Lib |
| 因為使用微服務，所以可以直接在Gradle導入    範例暫時使用2.1.3.RELEASE 版本。  如果非使用微服務，可導入以下Lib |

|  |
| --- |
| 2. 建立SpringSecurity Config類別 |
| 作為一個設定類別，要加上@Configuration  啟動SpringSecurity類別要加上@EnableWebSecurity  繼承WebSecutiryConfigurerAdapter 橋接器 |

|  |
| --- |
| 2.1 建立SpringSecurity Config類別 => 配置身分驗證管理器 |
| 當我們繼承了了WebSecutiryConfigurerAdapter 可以Overrid    配置我們要如何管理身分的驗證。  我們要配置存放這些身分的資訊在記憶體當中(可以存放在不同地方，之後會再研究)    配置編碼方式，Spring Secutiry5之後強制要求密碼需要進行加密，所以最基本的使用方式就是使用PasswrodEncoder 並且使用Spring 官方推薦的BCryptPasswordEncoder 進行加密。  /\*\* 配置密碼的編碼方式 spring secutiry5 之後強制要求密碼需要進行編碼 我們使用Spring 官方推薦的加密演算法方法BCryptPasswordEncoder \*/  PasswordEncoder pwdEncoder = new BCryptPasswordEncoder();  inMemoryUserDetailsManagerConfigurer.passwordEncoder(pwdEncoder);  我們叫建立使用者身分，該範例暫時把身份寫在程式裡。後續會根據不同情況取得身分。 |

|  |
| --- |
| 2.2 整個身分驗證配置大致上是這樣的… |
|  |

|  |
| --- |
| 2.3 配置Http攔截處理(基本做法後續會在研究) |
| 暫時先用以上的程式碼，該部分後續會再討論 |

|  |
| --- |
| 2.4 測試看看吧 |
| 當我們輸入了網址URL(再次聲明，我們使用SpringBoot並且建立了一個RESTController)  會出現Spring 官方提供的登入畫面。    登入後進會成功進入我們的RESTController的RequsetMapping!!! |

## 3使用Spring Security(身分驗證管理器-自定義使用者權限)

|  |
| --- |
| 1 建立SpringSecurity Config類別 => 配置身分驗證管理器(自定義使用者權限) |
| 當我們繼承了了WebSecutiryConfigurerAdapter 可以Override    配置我們要如何管理身分的驗證。  我們要配置存放這些身分的資訊在UserDetail當中，所以可以使用UserDetailsService()，並且該Service會回傳UserDetails這個類別。    當我們建立UserDetails的時候要傳入使用者的名稱(username)並且傳入密碼與權限範圍。  而權限範圍可以是一個Array。  我們也可以自定義使用者權限 |

## 99其他語法與效果

|  |
| --- |
| 1 預設情況之下不啟動Spring Security (暫時沒用，之後研究) |
| 語法為:security.basic.enabled: false |