3.Spring Boot 概述

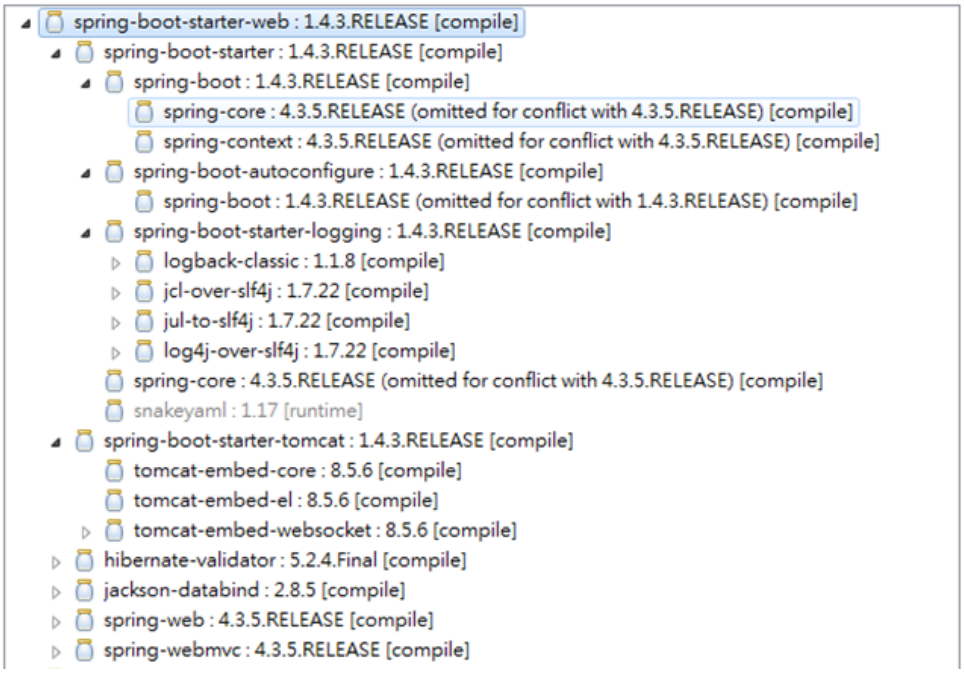
什麼是Spring Boot

- Spring Boot 並不是一個全新的框架，而是將已有的 Spring 組件整合起來。特點是去掉了繁瑣的 XML 配置，改使用約定或註解。所以熟悉了 Spring Boot 之後，開發效率將會提升一個檔次。
- 採約定優於配置的做法，特點是簡單、快速、便捷，但是這是建立在程式設計師熟悉這些約定的前提上。而 Spring 擁有一個龐大的生態體系，剛開始轉到 Spring Boot 完全捨棄 XML 時肯定是不習慣的，所以也會造成一些困擾。

- Spring Boot 是用來 讓 (參考說明 http://www.jianshu.com/p/d24bceea7665)
    - 所有的 Spring 開發者更快的入門
    - 開箱即用 ， 提供各種預設配置來簡化專案的設定
    - 內嵌 Web 容器 ， 簡化 Web 專案
    - 沒有冗餘代碼產生 ， 以及 XML 設定的要求

Spring Boot 核心功能

- 獨立運行的Spring 專案
    - Spring Boot 可以以jar 檔的形式獨立運行，運行一個Spring Boot 專案只需通過 java -jar xx.jar 來運行。
- 內嵌Web容器
    - Spring Boot 可選澤內嵌Tomcat、Jetty或者Undertow，這樣就無須以war檔形式部署web project。
- 提供starter簡化Maven 配置
    - Spring 提供了一系列的starter pom來簡化Maven的依賴加載；例如，當使用了spring-boot-starter-web時，會自動加入以下的libraries：



- 自動配置Spring而無需xml配置
    - Spring Boot會根據在class path中的jar檔、classes，為jar檔裏的class自動配置常用的Bean，這樣會大幅地減少開發人員所需的配置。當然，若實際開發中需要自行定義的Bean，也可以自己定義自動配置。而Spring Boot建議使用Java配置，不需使用任何xml配置即可實現 Spring 所有的配置。
- 應用系統監控
    - Spring Boot提供基於http、ssh、telnet對運行時的專案進行監控。

Spring Boot 目前提供的starter

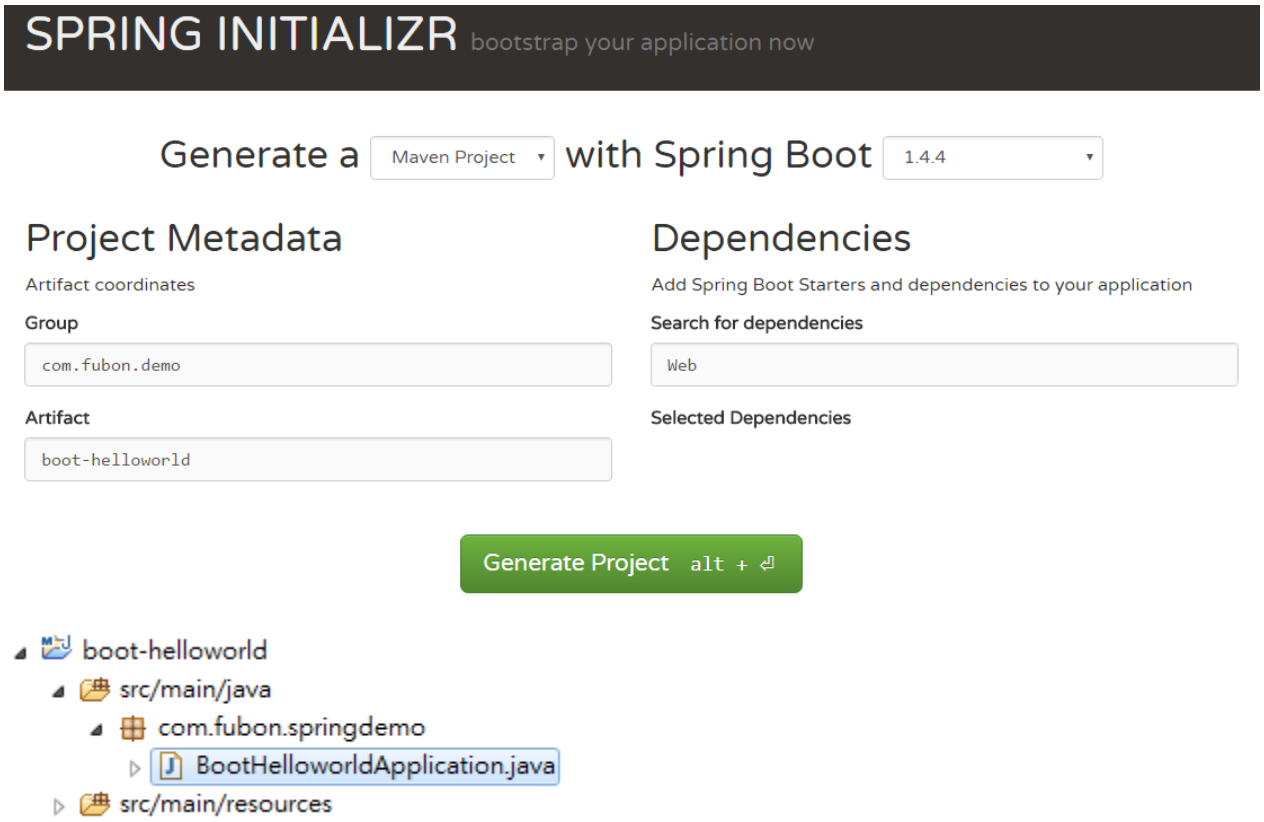
https://start.spring.io/

建立Spring Boot專案的方式

建立Spring.Boot.專案的方式

示例

- 使用從start.spring.io下載的方法新建 Spring Boot 專案後，建立的專案的 src 目錄下會有一個以 artifactId+Application 命名規則的入口 class。



- 因簡單示例，我們直接在入口程式編寫：
  - BootHelloworldApplication

BootHelloworldApplication

```
package com.fubon.springdemo;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

@SpringBootApplication  //1
@RestController//2
public class BootHelloworldApplication {

	public static void main(String[] args) {  //3
		SpringApplication.run(BootHelloworldApplication.class args);
	}

	@RequestMapping("/")
	String index() {
		return "Hello Spring Boot!";
	}
}
```

  - 說明：

    @SpringBootApplication 是Spring Boot專案的核心注解，主要目的是開啟自動配置，後續有較詳細的說明。

    請看Spring MVC 常見注解說明。

    Spring Boot 以標準的Java 類別的main方法，作為 Spring Boot 專案的啟動後的入口點， 後續有較詳細的說明。
  - Spring MVC 常見注解說明：

@Controller 標注在class上，表明這個class是Spring MVC裡的Controller，將其聲明為Spring 的一個Bean。 Dispatcher Servlet 會自動掃描標注了此注解的class，並將Web 請求 mapping 到標注了 @RequestMapping 的方法上。要特別指出的是，在聲明普通Bean的時候，使用 @Component、@Service、@Repository和@Controller 都是等同的，因為@Service、@Repository和@Controller都組合了@Component注解；但在 Spring MVC裡Dispatcher Servlet只會針對@Controller做掃描。

@RequestMapping 用來mapping Web請求(含訪問路徑和參數)與負責處理該請求的class 和method的。 @RequestMapping可標注在class和method上，標注在method上的路徑會繼承標注在class上的路徑。@RequestMapping支援Servlet的request 和response作為參數，也支援對request和response的媒體類型進行配置。

@RequestBody 表明request的參數是在request body中，而不是直接在URL的? 後面。此注解標注在method的接收參數前。

@ResponseBody 表明將method的返回值放在 response body中，而不是返回一個頁面。在很多基於Ajax的請求，可以以此注解返回資料而不是整個頁面；此注解可標注在返回值前或方法上。

@PathVariable 用來接收路徑參數， 此注解標注在參數前。

@RestController 是一個組合注解，組合了@Controller和@ResponseBody，這意味著若要設計一個返回值是放在response body的controller時，使用此注解是較直覺的。若沒有用此注解，要實現上述功能，則需自己在程式碼中加@Controller 和 @ResonseBody兩個注解。

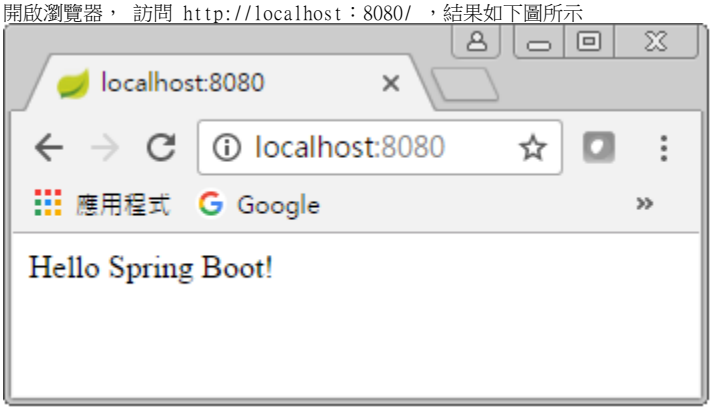
- 程式執行
  - 有３種方法可以Spring Boot專案：
  - main 方法執行：點選BootHelloworldApplication，以 [Run As] Spring Boot App 或 Java Application方式執行專案
  - maven命令執行：切換到 command line 模式， 在專案目錄下以Maven 命令方式執行

```
mvn spring-boot:run
```

```
D:\>cd d:\Workspace\boot-helloworld
D:\Workspace\boot-helloworld>mvn spring-boot:run
```

  - jar檔方式執行： 先用maven打包成jar檔，再執行 java -jar xx.jar

- 開啟瀏覽器， 訪問 http://localhost：8080/ ，結果如下圖所示



- 說明：
  - 本示例專案中，沒有WEB-INF目錄，也沒部署到任何Web Application Server上
  - 以 jar 檔方式打包後執行，就能接受http request並response 結果
  - Spring Boot Web starter 預設以tomcat 作為內嵌Web 容器， 以8080 作為接聽port