Spring Boot Client

Hello world example

Tạo 1 ứng dụng đơn giản sử dụng Spirng Boot + MVC + Rest based application

Bước 1: tạo thư mục gốc

Tạo 1 thư mục FirstApplication trong E:\TestFolder

Bước 2: tạo file gốc

Tạo file FirstApplication.groovy trong E:\TestFolder với code như sau-

@RestController

class FirstApplication {

@RequestMapping("/")

String welcome() {

"Welcome to TutorialsPoint.Com"

}

}

Bước 3: chạy ứng dụng

Dùng lệnh-

E:/Test/> spring run FirstApplication.groovy

Spring Boot CLI sẽ được thực thi, tải các dependency tương ứng, chạy embedded Tomcat, triển khai ứng dụng và chạy nó. Đầu ra ở console sẽ như sau-

Resolving dependencies..........................................................

................................................................................

........................

. \_\_\_\_ \_ \_\_ \_ \_

/\\ / \_\_\_'\_ \_\_ \_ \_(\_)\_ \_\_ \_\_ \_ \ \ \ \

( ( )\\_\_\_ | '\_ | '\_| | '\_ \/ \_> | \ \ \ \

\\/ \_\_\_)| |\_)| | | | | || (\_| | ) ) ) )

' |\_\_\_\_| .\_\_|\_| |\_|\_| |\_\\_\_, | / / / /

=========|\_|==============|\_\_\_/=/\_/\_/\_/

:: Spring Boot :: (v1.5.8.RELEASE)

2017-11-07 17:36:55.703 INFO 5528 --- [ runner-0] o.s.boot.SpringApplication:

Starting application on ...

2017-11-07 17:36:55.707 INFO 5528 --- [ runner-0] o.s.boot.SpringApplication:

No active profile set, falling back to default profiles: default

2017-11-07 17:36:56.067 INFO 5528 --- [ runner-0] ationConfigEmbeddedWebApplicationContext : Refreshing org.springframework.boot.context.embedded.AnnotationConfigEmbeddedWebApplicationContext@4c108392: startup date [Tue Nov 07 17:36:

56 IST 2017]; root of context hierarchy

2017-11-07 17:36:57.327 INFO 5528 --- [ runner-0] s.b.c.e.t.TomcatEmbeddedServletContainer : Tomcat initialized with port(s): 8080 (http)

2017-11-07 17:36:57.346 INFO 5528 --- [ runner-0] o.apache.catalina.core.StandardService : Starting service [Tomcat]

2017-11-07 17:36:57.354 INFO 5528 --- [ runner-0] org.apache.catalina.core.StandardEngine : Starting Servlet Engine: Apache Tomcat/8.5.23

2017-11-07 17:36:57.537 INFO 5528 --- [ost-startStop-1] org.apache.catalina.loader.WebappLoader : Unknown loader org.springframework.boot.cli.compiler.ExtendedGroovyClassLoader$DefaultScopeParentClassLoader@41bfad4f class org.springframew

ork.boot.cli.compiler.ExtendedGroovyClassLoader$DefaultScopeParentClassLoader

2017-11-07 17:36:57.567 INFO 5528 --- [ost-startStop-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplicationContext

2017-11-07 17:36:57.567 INFO 5528 --- [ost-startStop-1] o.s.web.context.ContextLoader : Root WebApplicationContext: initialization completed in 1500 ms

2017-11-07 17:36:57.725 INFO 5528 --- [ost-startStop-1] o.s.b.w.servlet.ServletRegistrationBean : Mapping servlet: 'dispatcherServlet' to [/]

2017-11-07 17:36:57.730 INFO 5528 --- [ost-startStop-1] o.s.b.w.servlet.FilterRegistrationBean : Mapping filter: 'characterEncodingFilter' to: [/\*]

2017-11-07 17:36:57.730 INFO 5528 --- [ost-startStop-1] o.s.b.w.servlet.FilterRegistrationBean : Mapping filter: 'hiddenHttpMethodFilter' to: [/\*]

2017-11-07 17:36:57.730 INFO 5528 --- [ost-startStop-1] o.s.b.w.servlet.FilterRegistrationBean : Mapping filter: 'httpPutFormContentFilter' to: [/\*]

2017-11-07 17:36:57.730 INFO 5528 --- [ost-startStop-1] o.s.b.w.servlet.FilterRegistrationBean : Mapping filter: 'requestContextFilter' to: [/\*]

2017-11-07 17:36:58.012 INFO 5528 --- [ runner-0] s.w.s.m.m.a.RequestMappingHandlerAdapter : Looking for @ControllerAdvice: org.springframework.boot.context.embedded.AnnotationConfigEmbeddedWebApplicationContext@4c108392: startup date [Tue Nov 07 17:36:56 IST 2017]; root of context hierarchy

2017-11-07 17:36:58.066 INFO 5528 --- [ runner-0] s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped "{[/]}" onto public java.lang.String FirstApplication.home()

2017-11-07 17:36:58.070 INFO 5528 --- [ runner-0] s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped "{[/error]}" onto public org.springframework.http.ResponseEntity<java.util.map> org.springframework.boot.autoconfigure.web.BasicErrorController.error(javax.servlet.http.HttpServletRequest)

2017-11-07 17:36:58.071 INFO 5528 --- [ runner-0] s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped "{[/error],produces=[text/html]}" onto public org.springframework.web.servlet.ModelAndView org.springframework.boot.autoconfigure.web

.BasicErrorController.errorHtml(javax.servlet.http.HttpServletRequest,javax.servlet.http.HttpServletResponse)

2017-11-07 17:36:58.096 INFO 5528 --- [ runner-0] o.s.w.s.handler.SimpleUrlHandlerMapping : Mapped URL path [/webjars/\*\*] onto handler of type [class org.springframework.web.servlet.resource.ResourceHttpRequestHandler]

2017-11-07 17:36:58.096 INFO 5528 --- [ runner-0] o.s.w.s.handler.SimpleUrlHandlerMapping : Mapped URL path [/\*\*] onto handler of type [class org.springframework.web.servlet.resource.ResourceHttpRequestHandler]

2017-11-07 17:36:58.129 INFO 5528 --- [ runner-0] o.s.w.s.handler.SimpleUrlHandlerMapping : Mapped URL path [/\*\*/favicon.ico] onto handler of type [class org.springframework.web.servlet.resource.ResourceHttpRequestHandler]

2017-11-07 17:36:58.626 INFO 5528 --- [ runner-0] o.s.j.e.a.AnnotationMBeanExporter : Registering beans for JMX exposure on startup

2017-11-07 17:36:58.696 INFO 5528 --- [ runner-0] s.b.c.e.t.TomcatEmbeddedServletContainer : Tomcat started on port(s): 8080 (http)

2017-11-07 17:36:58.699 INFO 5528 --- [ runner-0] o.s.boot.SpringApplication : Started application in 3.529 seconds (JVM running for 190.196)

2017-11-07 17:37:20.217 INFO 5528 --- [nio-8080-exec-1] o.a.c.c.C.[Tomcat].[localhost].[/] : Initializing Spring FrameworkServlet 'dispatcherServlet'

2017-11-07 17:37:20.218 INFO 5528 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : FrameworkServlet 'dispatcherServlet': initialization started

2017-11-07 17:37:20.238 INFO 5528 --- [nio-8080-exec-1] o.s.web.servlet.DispatcherServlet : FrameworkServlet 'dispatcherServlet': initialization completed in 20 ms

</java.util.map

Bước 4: chạy ứng dụng trên trình duyệt

Chạy URL "**http://localhost:8080/**" và bạn sẽ thấy kết quả-

Hello World

Điểm lưu ý các điểm sau để hiểu cách hoạt động của SB CLI-

* Tất cả các JAR dependency đều chỉ tải 1 lần duy 1 lần đầu tiên
* Spring CLI tự quét tìm tất cả JAR dependency cần tải dựa trên code các class, annotation mà bạn dùng.
* Việc triển khai file war và chạy tomcat đều trên cổng mặc định 8080.

Grab Dependency Deduction

Kho code Standard Groovy bao gồm cả @Grab annotation cho phép khai báo các dependency từ thư viện của ên thứ ba. Sử dụng @Grab annotation, Grape Dependency Manager tải jar tương tự như Maven/Gradle mà không cần phải xây dựng công cụ. Spring Boot sẽ cố suy ra các thư viện cần thiết dựa trên code của bạn. vd, dùng @RestController có nghĩa là sẽ lấy thư viện “Tomcat” và “Spring MVC”.

## Grab Hints

|  |  |
| --- | --- |
| **Sr.No.** | **Hint & Dependency to Download/Link** |
| 1 | **JdbcTemplate, NamedParameterJdbcTemplate, DataSource**  JDBC Application |
| 2 | **@EnableJms**  JMS Application |
| 3 | **@EnableCaching**  Caching abstraction |
| 4 | **@Test**  JUnit |
| 5 | **@EnableRabbit**  RabbitMQ |
| 6 | **@EnableReactor**  Project Reactor |
| 7 | **extends Specification**  Spock test |
| 8 | **@EnableBatchProcessing**  Spring Batch |
| 9 | **@MessageEndpoint, @EnableIntegrationPatterns**  Spring Integration |
| 10 | **@EnableDeviceResolver**  Spring Mobile |
| 11 | **@Controller, @RestController, @EnableWebMvc**  Spring MVC + Embedded Tomcat |
| 12 | **@EnableWebSecurity**  Spring Security |
| 13 | **@EnableTransactionManagement**  Spring Transaction Management |

Default Statement

## Default Imports

Spring CLI tự động nhập nhiều thư viện theo mặc định nên không cần nhập rõ ràng. Bây giờ chúng ta hãy xem một Groovy script sau để hiểu Default Imports.

@RestController

class FirstApplication {

@RequestMapping("/")

String welcome() {

"Welcome to TutorialsPoint.Com"

}

}

Tại đây các import cho @RestController, @RequestMapping annotations đã được thêm sẵn mặc định bởi Spring Boot. Chúng ta thậm chí không cần dùng đầy đủ tên điều kiện. bạn có thể kiểm tra bằng cách chạy ứng dụng.

Dùng lệnh-

E:/Test/> spring run FirstApplication.groovy

Lệnh trên sẽ cho kết quả ở màn hình console như vầy −

. \_\_\_\_ \_ \_\_ \_ \_

/\\ / \_\_\_'\_ \_\_ \_ \_(\_)\_ \_\_ \_\_ \_ \ \ \ \

( ( )\\_\_\_ | '\_ | '\_| | '\_ \/ \_> | \ \ \ \

\\/ \_\_\_)| |\_)| | | | | || (\_| | ) ) ) )

' |\_\_\_\_| .\_\_|\_| |\_|\_| |\_\\_\_, | / / / /

=========|\_|==============|\_\_\_/=/\_/\_/\_/

:: Spring Boot :: (v1.5.8.RELEASE)

...

2017-11-07 19:22:17.310 INFO 4824 --- [ runner-0] o.s.boot.SpringApplication

: Started application in 3.405 seconds (JVM running for 7.021)

## Automatic Main Method

Chúng ta không cần phải tạo một phương hàm main như thông thường cho groovy script để khởi tạo ứng dụng Spring, nó sẽ được tạo sẵn 1 cách tự động cho ứng dụng Spring boot.

Thymeleaf Project

Phần này chỉ chúng ta cách tạo 1 project Thymeleaf đơn giản để mô tả các công dụng của Spring CLI. Các bước thực hiện-

|  |  |
| --- | --- |
| **Sr.No** | **Step & Description** |
| 1 | Tạo 1 thư mục với tên *TestApplication*  và các thư mục con :  *templates* và *static* |
| 2 | Tạo *message.groovy* trong thư mục *TestApplication* , *message.html* trong thư mục templates, , *index.html* trong thư mục *static* với nội dung như bên dưới. |
| 3 | Biên dịch và chạy ứng dụng để xác minh kết quả của logic đã triển khai. |

### TestApplication/message.groovy

@Controller

@Grab('spring-boot-starter-thymeleaf')

class MessageController {

@RequestMapping("/message")

String getMessage(Model model) {

String message = "Welcome to TutorialsPoint.Com!";

model.addAttribute("message", message);

return "message";

}

}

### TestApplication/templates/message.html

<!DOCTYPE HTML>

<html xmlns:th = "http://www.thymeleaf.org">

<head>

<title>Spring Boot CLI Example</title>

<meta http-equiv = "Content-Type" content = "text/html; charset = UTF-8" />

</head>

<body>

<p th:text = "'Message: ' + ${message}" />

</body>

</html>

### TestApplication/static/index.html

<!DOCTYPE HTML>

<html>

<head>

<title>Spring Boot CLI Example</title>

<meta http-equiv = "Content-Type" content = "text/html; charset = UTF-8" />

</head>

<body>

<p>Go to <a href = "/msg">Message</a></p>

</body>

</html>

## Run the application

Chạy ứng dụng, sử dụng lệnh −

E:/Test/TestApplication/> spring run \*.groovy

Spring Boot CLI sẽ thực hiện hành động, tải dependency cần thiết chạy tomcat, triển khai ứng dụng và bắt đầu. kết quả trên cửa sổ console-

Resolving dependencies.............................

. \_\_\_\_ \_ \_\_ \_ \_

/\\ / \_\_\_'\_ \_\_ \_ \_(\_)\_ \_\_ \_\_ \_ \ \ \ \

( ( )\\_\_\_ | '\_ | '\_| | '\_ \/ \_> | \ \ \ \

\\/ \_\_\_)| |\_)| | | | | || (\_| | ) ) ) )

' |\_\_\_\_| .\_\_|\_| |\_|\_| |\_\\_\_, | / / / /

=========|\_|==============|\_\_\_/=/\_/\_/\_/

:: Spring Boot :: (v1.5.8.RELEASE)

...

2017-11-08 16:27:28.300 INFO 8360 --- [ runner-0] s.b.c.e.t.TomcatEmbeddedServletContainer : Tomcat started on port(s): 8080 (http)

2017-11-08 16:27:28.305 INFO 8360 --- [ runner-0] o.s.boot.SpringApplication : Started application in 4.203 seconds (JVM running for 38.792)

## Browse the application in Browser

Spring dựa trên ứng dụng Rest đã sẵn sằng. Mở url "http: // localhost: 8080 /" và bạn sẽ thấy đầu ra sau

Go to Message

Nhấn vào link Message và bạn sẽ thấy−

Message − Welcome to TutorialsPoint.Com!

## Important points

Các hoạt động Spring CLI thực hiện-

* @Grab('spring-boot-starter-thymeleaf') annotation chỉ định CLI tải spring-boot-starter-thymeleaf 1.5.8.RELEASE version.
* Spring CLI tự động phát hiện ra phiên bản bằng cách sử dụng metadata của nó dù chúng ta không chỉ định bất cứ groud id hay version id nào.
* Cổng mặc định vẫn là 8080

Testing application

Tạo project Hello World Example. Các bước −

|  |  |
| --- | --- |
| **Sr.No** | **Step & Description** |
| 1 | Tạo *FirstApplication.groovy* và *TestFirstApplication.groovy* trong thư mục *Test* nội dung bên dưới. |
| 2 | Biên dịch và chạy ứng dụng để xác minh kết quả của logic đã triển khai. |

### FirstApplication/FirstApplication.groovy

@RestController

class FirstApplication {

@RequestMapping("/")

String welcome() {

"Welcome to TutorialsPoint.Com"

}

}

### FirstApplication/TestFirstApplication.groovy

class TestFirstApplication {

@Test

void welcomeTest() {

assertEquals("Welcome to TutorialsPoint.Com", new FirstApplication().welcome())

}

}

## Run the application

Để chạy ứng dụng dùng lệnh −

E:/Test/FirstApplication/> spring test FirstApplication.groovy TestFirstApplication.groovy

Kết quả trên console

Resolving dependencies........................................................

.

Time: 0.457

OK (1 test)

## Important points

Consider the following points to understand the actions taken by Spring CLI −

* The @Test annotation directs CLI to download JUnit 4.12 version.
* Spring CLI automatically detects the version using its metadata, as we have not specified any dependency.
* Finally, after code compilation, test the application.

Packaging application

Spring boot CLI cung cấp lệnh jar để đóng gói ứng dụng thành như 1 file jartạo thử 1 project [Starter Thymeleaf Project](https://www.tutorialspoint.com/springbootcli/springbootcli_thymeleaf_project.htm) Chapter để thể hiện tính đóng gói của Spring Boot

Các bước-

Package the application

Để đóng gói ứng dụng, hãy bắt đầu bằng cách gõ lệnh sau −

E:/Test/TestApplication/> spring jar TestApplication.jar \*.groovy

Output

Lệnh sẽ in đầu ra sau −

E:/Test/TestApplication/> spring jar TestApplication.jar \*.groovy

Output

Bây giờ bạn có thể thấy hai tệp mới được tạo trong thư mục.

* TestApplication.**TestApplication.jar** − An executable jar file.
* **TestApplication.jar.original** − Original jar file.

Include/Exclude

Theo mặc định, các thư mục sau được bao gồm cùng với nội dung của chúng −

* public
* resources
* static
* templates
* META-INF

Theo mặc định, các thư mục sau được loại trừ cùng với nội dung của chúng

* build
* target
* \*.jar files
* \*.groovy files

Sử dụng - include, ta có thể bao gồm các thư mục loại trừ. Sử dụng --exclude, ta có thể loại trừ các thư mục đi kèm.

Running the Executable Jar

Để chạy Jar thực thi, gõ lệnh sau −

E:/Test/TestApplication/> java -jar TestApplication.jar

Lệnh trên sẽ tạo đầu ra sau trên bàn điều khiển −

. \_\_\_\_ \_ \_\_ \_ \_

/\\ / \_\_\_'\_ \_\_ \_ \_(\_)\_ \_\_ \_\_ \_ \ \ \ \

( ( )\\_\_\_ | '\_ | '\_| | '\_ \/ \_> | \ \ \ \

\\/ \_\_\_)| |\_)| | | | | || (\_| | ) ) ) )

' |\_\_\_\_| .\_\_|\_| |\_|\_| |\_\\_\_, | / / / /

=========|\_|==============|\_\_\_/=/\_/\_/\_/

:: Spring Boot :: (v1.5.8.RELEASE)

...

2017-11-08 16:27:28.300 INFO 8360 --- [ runner-0] s.b.c.e.t.TomcatEmbeddedServletContainer : Tomcat started on port(s): 8080 (http)

2017-11-08 16:27:28.305 INFO 8360 --- [ runner-0] o.s.boot.SpringApplication : Started application in 4.203 seconds (JVM running for 38.792)

Browse the application in Browser

Mở url "http: // localhost: 8080 /" và bạn sẽ thấy đầu ra sau –

Go to Message

Nhấp vào Message và bạn sẽ thấy đầu ra sau đây −

Message: Welcome to TutorialsPoint.Com!

Creating Project

Spring Boot CLI có thể được sử dụng để tạo một dự án mới với maven là công cụ xây dựng mặc định bằng lệnh init. Maven sẽ sử dụng dịch vụ https://start.spring.io. Trong ví dụ sau, chúng ta sẽ tạo một ứng dụng web bằng thymeleaf.

Chuyển đến thư mục E: \ Test và gõ lệnh sau -

E:/Test> spring init --dependencies = web,thymeleaf MavenApplication.zip

Lệnh trên sẽ tạo đầu ra sau −

Using service at https://start.spring.io

Content saved to MavenApplication.zip

Using shell

Spring Boot CLI cung cấp giao diện Shell để chạy các lệnh trong đó, chúng ta có thể chạy trực tiếp các lệnh như dưới đây.

Chuyển đến thư mục E: \ Test và gõ lệnh sau –

E:/Test> spring shell

Lệnh trên sẽ tạo đầu ra sau −

?[1mSpring Boot?[m?[2m (v1.5.8.RELEASE)?[m

Hit TAB to complete. Type 'help' and hit RETURN for help, and 'exit' to quit.

## Running commands in Shell

Trong phần này, chúng ta sẽ tìm hiểu cách chạy các lệnh trong Shell. Nhập nội dung sau và xem đầu ra −

version

Spring CLI v1.5.8.RELEASE

## Bạn có thể nhấn tab để tự động hoàn thành các lệnh và gõ exit để hoàn thành shell console.

## Testing the application in shell

Bây giờ chúng ta hãy tìm hiểu làm thế nào để kiểm tra ứng dụng trong shell. Nhập dòng code sau đây và xem đầu ra –

E:\Test\FirstApplication>spring shell

?[1mSpring Boot?[m?[2m (v1.5.8.RELEASE)?[m

Hit TAB to complete. Type 'help' and hit RETURN for help, and 'exit' to quit.

$ test FirstApplication.groovy TestFirstApplication.groovy

.

Time: 0.347

OK (1 test)

$ exit

E:\Test\FirstApplication>