

in28minutes

Spring Boot 2.0.0.RELEASE Upgrade

Get upto date with the most popular Java
framework to develop awesome micro
services!



Table of Contents

1. [Spring Boot Version Details](#)
2. [How are Spring Releases Versioned?](#)
3. [Release Notes Extracts](#)
4. [Course Project Updates](#)
5. [Course Sections Updated](#)
6. [New Videos Created/Updated](#)
7. [Patch](#)
8. [Congratulations](#)
9. [Keep Learning in 28 Minutes](#)

Spring Boot Version Details

Previous Versions

- Spring Boot - 2.0.0.M3
- Spring Cloud - Finchley.M2

Current Versions

- Spring Boot - 2.0.0.RELEASE
- Spring Cloud - Finchley.M8

How are Spring Releases Versioned?

Take a look at <https://github.com/spring-projects/spring-boot/wiki/Spring-Boot-2.-0.0-RC2-Release-Notes>

- v2.0.0 M1 - May 16, 2017 - First Milestone
- v2.0.0 M2
- v2.0.0 M3 - July 2017 - Most of our courses
- v2.0.0 M4
- v2.0.0 M5
- v2.0.0 M6
- v2.0.0 M7
- v2.0.0 RC1 - Jan 31 2018 - First Release Candidate
- v2.0.0 RC2 - Feb 21 2018
- v2.0.0 RELEASE - March 01 2018

Release Notes Extracts

Following notes are extract from different release notes of Spring Boot 2.0.0 Releases.

6 Releases - All links available at - <https://github.com/spring-projects/spring-boot/wiki/Spring-Boot-2.0.0-RC2-Release-Notes>

Details - Actuator Upgrade - <https://github.com/spring-projects/spring-boot/wiki/Spring-Boot-2.0.0-M4-Release-Notes#build>

Actuator

Already covered at large in the migration part, Actuator now runs natively on Spring MVC, Spring WebFlux and Jersey. Adding the dependency and enabling web endpoints (using for instance `endpoints.default.web.enabled`) is all that's required.

- Security Upgrade - <https://github.com/spring-projects/spring-boot/wiki/Spring-Boot-2.0.0-M4-Release-Notes#security>

Another important change is that there is no longer a separate security auto-configuration for the Actuator: (`management.security.enabled`) no longer exists and web endpoints are disabled by default (`endpoints.default.web.enabled`) to prevent exposing sensitive information by default. The sensitive flag of each endpoint is also gone to make things more explicit in the security configuration. To restore the previous behaviour:

Enable Actuator web endpoints:

`endpoints.default.web.enabled=true` in your configuration

Create or adapt your security configuration to secure

endpoints with the role of your choice (see the Security section

below)

M5 Management Server-related properties have been moved from `management.*` to `management.server.*`. We also fixed the meaning of `management.server.context-path`: it is now the

endpoint management equivalent of `server.context-path` (only active when `management.server.port` is set). Additionally, you can also set the base path for the management endpoints with a new, separate property: `management.endpoints.web.base-path`.

For example, if you've set `management.server.context-path=/management` and `management.endpoints.web.base-path=/application`, you'll be able to reach the health endpoint at the following path:
`/management/application/health`.

M6

After some feedback from the community, the default `management.endpoints.web.base-path` has been changed from `"/application"` to `"/actuator"` to avoid collision with user-defined mappings (see #10970).

The `endpoints.` keys have moved to `management.endpoints`.

RC1

Spring Security default user

We've restored the ability to auto-configure a single user via configuration keys. These can be found at `spring.security.user`.

- OAuth 2.0 Support

Functionality from the Spring Security OAuth project is being migrated to core Spring Security. OAuth 2.0 client support has already been added and additional features will be migrated in due course.

If you depend on Spring Security OAuth features that have not yet been migrated you will need to add `org.springframework.security.oauth:spring-security-oauth2` and configure things manually. If you only need OAuth 2.0 client support you can use the auto-configuration provided by Spring Boot 2.0. We're also continuing to support Spring Boot 1.5 so

older applications can continue to use that until an upgrade path is provided.

- Jackson serialization

The default value for `spring.jackson.serialization.write-dates-as-timestamps` is now set to `false`, meaning all JSR310 date types will be serialized as ISO-8601 strings instead of array-like types.

RC1

Jackson

`SerializationFeature.WRITE_DATES_AS_TIMESTAMPS` is now disabled by default.

Date conversion

It is possible to control the date format for date types from well-known packages (`java.util`, `org.joda.time` and `java.time`) using a single property.

- Renamed

```
=====
CONDITION EVALUATION DELTA
=====
```

Spring Cloud

<https://github.com/spring-projects/spring-cloud/wiki/Spring-Cloud-Finchley-Release-Notes>

- Remove the Zipkin Stream server <https://github.com/spring-cloud/spring-cloud-sleuth/issues/727>
- All of Feign code from Spring Cloud Netflix has been moved to a new project, Spring Cloud OpenFeign
- spring-cloud-starter-eureka is deprecated. It is now called
- spring-cloud-starter-netflix-eureka-client

Java 9 vs Java 8

```
<properties>
  <!-- Generic properties -->
  <java.version>9</java.version>
</properties>
```

java.lang.NoClassDefFoundError: javax/xml/bind/JAXBException

Affects all modules using JPA

Hibernate typically requires JAXB that's no longer provided by default. You either need to add a dependency to your project:

```
<dependency>
  <groupId>javax.xml.bind</groupId>
  <artifactId>jaxb-api</artifactId>
</dependency>
```


Course Project Updates

REST Web Services

Actuator

Replace old url - <http://localhost:8080/application> New URL

- <http://localhost:8080/actuator>

application.properties

```
management.endpoints.web.exposure.include=*
```

In HAL Browser, enter the actuator URL to browse.

Spring Security

```
#security.user.name=username
#security.user.password=password
spring.security.user.name=username
spring.security.user.password=password

+         auth.inMemoryAuthentication()
+
.passwordEncoder(NoOpPasswordEncoder.getInstance())
+         .withUser("in28Minutes").password("dummy")
```

Internationalization!

```
@GetMapping("/hello")
public String helloWorld() {
    return msgSource.getMessage("msg.hello", null,
```

```

    "Whoops!", LocaleContextHolder.getLocale());
    }

    @Bean
    public LocaleResolver localeResolver() {
        AcceptHeaderLocaleResolver localeResolver = new
AcceptHeaderLocaleResolver();
        localeResolver.setDefaultLocale(Locale.US);
        return localeResolver;
    }

spring.jackson.serialization.write-dates-as-
timestamps=false
spring.messages.basename=messages

```

Spring Cloud Microservices

Artifact/Code Changes

```

-    <artifactId>spring-cloud-starter-zuul</artifactId>
+    <artifactId>spring-cloud-starter-netflix-
zuul</artifactId>

-    <artifactId>spring-cloud-starter-feign</artifactId>
+    <artifactId>spring-cloud-starter-
openfeign</artifactId>

-    <artifactId>spring-cloud-starter-eureka</artifactId>
+    <artifactId>spring-cloud-starter-netflix-eureka-
client</artifactId>
-

```

```

    <artifactId>spring-cloud-starter-eureka-
server</artifactId>

+

    <artifactId>spring-cloud-starter-netflix-eureka-server</
-      <artifactId>spring-cloud-starter-ribbon</artifactId>
+      <artifactId>spring-cloud-starter-netflix-
ribbon</artifactId>

-      <artifactId>spring-cloud-starter-hystrix</artifactId>

+      <artifactId>spring-cloud-starter-netflix-
hystrix</artifactId>

- public AlwaysSampler defaultSampler() {
-     return new AlwaysSampler();
+ public Sampler defaultSampler() {
+     return Sampler.ALWAYS_SAMPLE;

```

ZipkinUi, ZipkinStream, StreamRabbit Not Available

Not available on spring.io for current version of spring boot.

New Installation Approach for Zipkin

Quick Start Page - <https://zipkin.io/pages/quickstart>

Downloading Zipkin Jar - https://search.maven.org/remote_content?g=io.zipkin.java&a=zipkin-server&v=LATEST&c=exec

Command to run

```

RABBIT_URI=amqp://localhost java -jar zipkin-server-2.5.2-
exec.jar

```

Course Sections Updated

- 35.SpringWebServices
 - 02.IntroductionToSpringFramework
 - 03.IntroductionToSpringBootFramework
 - 05.BasicsOfRestfulWebServices
 - 06.AdvancedFeatures-RestfulWebServices
 - 08.ConnectingRestfulServiceToJPA
- 36.SpringBootMasterClass (TODO)
 - 05.Spring-Boot-Advanced
 - 07.Connecting-Spring-Boot-Web-Application-To-JPA
 - 09.Appendix.Spring-Introduction-In-10-Steps
- 37.SpringMasterClass
 - 01.IntroductionToSpringFramework
 - 02.SpringCoreInDepth
 - 06.IntroductionToSpringBootFramework
 - 08.Spring-JDBC-And-Upgrade-To-JPA
- 39.HibernateWithJPAandSpringBoot
 - 01.IntroductionToSpringBootFramework
 - 02.JourneyFromSpringJDBCToJPA
 - 04.JPA-In-Depth
 - 09.Appendix.IntroductionToSpringFramework
- 40.SpringMicroservices
 - 02.RestfulWebServices
 - 03.MicroservicesWithSpringCloud
 - 99.01.APPENDIX.IntroductionToSpringBootFramework

New Videos Created/Updated

- Spring Framework Intro
 - 01.IntroductionToSpringFramework-Step01-SettingUpASpringProjectUsingSpringInitializr
- Restful Web Services
 - 02.RestfulWebServices-Step-08—Implementing-GET-Methods-for-User-Resource
 - 02.RestfulWebServices-Step-18—Internationalization-for-RESTful-Services
 - 02.RestfulWebServices-Step-18—Internationalization-for-RESTful-Services-Improvements-Part2
 - 02.RestfulWebServices-Step-23—Monitoring-APIs-with-Spring-Boot-Actuator
 - 02.RestfulWebServices-Step-28—Implementing-Basic-Authentication-with-Spring-Security
 - 02.RestfulWebServices-Step-30—Creating-User-Entity-and-some-test-data
- Spring Core in Depth
 - 02.SpringCoreInDepth-Step-15—Complex-scenarios-with-Scope-of-a-Spring-Bean—Mix-of-Prototype-and-Singleton
- Microservices
 - 03.MicroservicesWithSpringCloud-Step-16—Configure-JPA-and-Initialized-Data
 - 03.MicroservicesWithSpringCloud-Step-21—Using-Feign-REST-Client-for-Service-Invocation
 - 03.MicroservicesWithSpringCloud-Step-22—Setting-up-client-side-load-balancing-with-Ribbon
 - 03.MicroservicesWithSpringCloud-Step-26—Connecting-Currency-Conversion-Microservice-to-Eureka
- Microservices (Contd...)

- ◦ 03.MicroservicesWithSpringCloud-Step-27—Connecting-Currency-Exchange-Microservice-to-Eureka
- 03.MicroservicesWithSpringCloud-Step-36—Implementing-Spring-Cloud-Sleuth
- 03.MicroservicesWithSpringCloud-Step-39—Setting-up-Distributed-Tracing-with-Zipkin
- 03.MicroservicesWithSpringCloud-Step-41---Using-Zipkin-UI-Dashboard-to-trace-requests

- JPA
 - 04.JPA-In-Depth-Step-59—Transaction-Management—ACID-Properties

- Spring Boot & Web Application
 - 06.IntroductionToSpringBootFramework-Step09-UsingSpringBootActuatorToMonitorYourApplications
 - 07.Connecting-Spring-Boot-Web-Application-To-JPA-Step-27—Configuring-H2-Console
 - 08.Spring-JDBC-And-Upgrade-To-JPA-Step-02—Launching-up-H2-Console
 - Step03-UsingSpringInitializrToCreateASpringBootApplication-20170924

Patch

Complete patch is available at

<https://github.com/in28minutes/spring-microservices/blob/master/spring-boot-2-0-0-Upgrade-notes.md#patch>

Congratulations

You have made a great choice in learning with in28Minutes. You are joining 150,000+ Learners learning everyday with us.

150,000+ Java beginners are learning from in28Minutes to become experts on APIs, Web Services and Microservices with Spring, Spring Boot and Spring Cloud.



Full Stack Developer with Javascript, Angular & React



Master Microservices with Spring Boot & Spring Cloud



Master Web Services and REST API with Spring Boot



Master Hibernate & JPA with Spring Boot in 100 Steps



Learn Spring Boot in 100 Steps - Beginner to Expert



Spring Master Class - Beginner to Expert in 100 Steps



Java Servlets and JSP - Build Java EE app in 25 Steps

in28minutes

Become an expert on Spring Boot, APIs, Microservices and Full Stack Development

[Checkout the Complete in28Minutes Course Guide](#)