Spring Boot 深度实践 – 系列总览

核心特性

组件自动装配

激活：`@EnableAutoConfiguration`

配置：`/META-INF/spring.factories`

实现：`XXXAutoConfiguration`

https://docs.spring.io/spring-boot/docs/2.0.1.RELEASE/reference/htmlsingle/#boot-features-spring-mvc-auto-configuration

Web MVC

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
</dependency>
```

嵌入式Web容器

Web Servlet 容器

Web Reactive 容器

生产准备特性

指标（Metrics）

健康检查（Health Check）

外部化配置（Externalized Configuration）

Web 应用

传统 Servlet 应用

依赖

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
</dependency>
```

Servlet 组件

- Servlet
  - 实现
    - `@WebServlet`
    - HttpServlet
    - 注册
  - URL 映射
    - `@WebServlet(urlPatterns = "/my/servlet")`
  - 注册
    - `@ServletComponentScan(basePackages = "com.imooc.diveinspringboot.web.servlet")`
- Filter
- Listener

Servlet 注册

Servlet 注解

- `@ServletComponentScan` +
  - `@WebServlet`
  - `@WebFilter`
  - `@WebListener`

Spring Bean

- `@Bean` +
  - Servlet
  - Filter
  - Listener

RegistrationBean

- `ServletRegistrationBean`
- `FilterRegistrationBean`
- `ServletListenerRegistrationBean`

异步非阻塞

异步 Servlet

- `javax.servlet.ServletRequest#startAsync()`
- `javax.servlet.AsyncContext`

非阻塞 Servlet

- `javax.servlet.ServletInputStream#setReadListener`
  - `javax.servlet.ReadListener`
- `javax.servlet.ServletOutputStream#setWriteListener`
  - `javax.servlet.WriteListener`

Spring Web MVC 应用

Web MVC 视图

- `ViewResolver`
- `View`

模板引擎

- Thymeleaf
- Freemarker
- JSP

内容协商

- `ContentNegotiationConfigurer`
- `ContentNegotiationStrategy`
- `ContentNegotiatingViewResolver`

异常处理

- `@ExceptionHandler`
- `HandlerExceptionResolver`
  - `ExceptionHandlerExceptionResolver`
- `BasicErrorController` (Spring Boot)

Web MVC REST

资源服务

- `@RequestMapping`
  - `@GetMapping`
- `@ResponseBody`
- `@RequestBody`

资源跨域

- `CrossOrigin`
- `WebMvcConfigurer#addCorsMappings`
- 传统解决方案
  - IFrame
  - JSONP

服务发现

- HATEOS

Web MVC 核心

核心架构

处理流程

核心组件

- `DispatcherServlet`
- `HandlerMapping`
- `HandlerAdapter`
- `ViewResolver`
- ...

Spring Web Flux 应用

Reactor 基础

Java Lambda

Mono

Flux

Web Flux 核心

Web MVC 注解兼容

- `@Controller`
- `@RequestMapping`
- `@ResponseBody`
- `@RequestBody`
- ...

函数式声明

- `RouterFunction`

异步非阻塞

- Servlet 3.1 +
- Netty Reactor

使用场景

页面渲染

REST 应用

性能测试

http://blog.ippon.tech/spring-5-webflux-performance-tests/

Web Server 应用

切换 Web Server

切换其他 Servlet 容器

- Tomcat -> Jetty

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-web</artifactId>
    <exclusions>
        <!-- Exclude the Tomcat dependency -->
        <exclusion>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-tomcat</artifactId>
        </exclusion>
    </exclusions>
</dependency>
<!-- Use Jetty instead -->
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-jetty</artifactId>
</dependency>
```

替换 Servlet 容器

- WebFlux

```
        <!--<dependency>-->
            <!--<groupId>org.springframework.boot</groupId>-->
            <!--<artifactId>spring-boot-starter-web</artifactId>-->
            <!--<exclusions>-->
                <!--&lt;!&ndash; Exclude the Tomcat dependency &ndash;&gt;-->
                <!--<exclusion>-->
```

```
                    <!--<groupId>org.springframework.boot</groupId>-->
                    <!--<artifactId>spring-boot-starter-tomcat</artifactId>-->
                <!--</exclusion>-->
            <!--</exclusions>-->
        <!--</dependency>-->

        <!--&lt;!&ndash; Use Jetty instead &ndash;&gt;-->
        <!--<dependency>-->
            <!--<groupId>org.springframework.boot</groupId>-->
            <!--<artifactId>spring-boot-starter-jetty</artifactId>-->
        <!--</dependency>-->

    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-webflux</artifactId>
    </dependency>
```

自定义 Servlet Web Server

- `WebServerFactoryCustomizer`

自定义 Reactive Web Server

- `ReactiveWebServerFactoryCustomizer`

数据相关

关系型数据

JDBC

依赖

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-jdbc</artifactId>
</dependency>
```

数据源

- `javax.sql.DataSource`

`JdbcTemplate`

自动装配

- `DataSourceAutoConfiguration`

JPA

依赖

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>
```

实体映射关系

- `@javax.persistence.OneToOne`
- `@javax.persistence.OneToMany`
- `@javax.persistence.ManyToOne`
- `@javax.persistence.ManyToMany`
- ...

实体操作

- `javax.persistence.EntityManager`

自动装配

- `HibernateJpaAutoConfiguration`

事务（Transaction）

依赖

```
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-tx</artifactId>
</dependency>
```

Spring 事务抽象

- `PlatformTransactionManager`

JDBC 事务处理

- `DataSourceTransactionManager`

自动装配

- `TransactionAutoConfiguration`

功能扩展

Spring Boot 应用

`SpringApplication`

失败分析

- `FailureAnalysisReporter`

应用特性

- `SpringApplication` Fluent API

Spring Boot 配置

- 外部化配置
  - `ConfigurationProperty`
- `@Profile`
- 配置属性
  - `PropertySources`

Spring Boot Starter

运维管理

Spring Boot Actuator

依赖

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-actuator</artifactId>
</dependency>
```

端点（Endpoints）

Web Endpoints

JMX Endpoints

健康检查（Health Checks）

`Health`

`HealthIndicator`

指标（Metrics）

内建 Metrics

- Web Endpoint：`/actuator/metrics`

自定义 Metrics