

Spring Boot 静态资源处理

原创

2015年12月27日 15:38:10

69717

11

19

Spring Boot 静态资源处理

Spring Boot 系列

1. Spring Boot 入门 (http://blog.csdn.net/isea533/article/details/50278205)
2. Spring Boot 属性配置和使用 (http://blog.csdn.net/isea533/article/details/50281151)
3. Spring Boot 集成MyBatis (http://blog.csdn.net/isea533/article/details/50359390)
4. Spring Boot 静态资源处理 (http://blog.csdn.net/isea533/article/details/50412212)
5. Spring Boot - 配置排序依赖技巧 (http://blog.csdn.net/isea533/article/details/53975720)
6. Spring Boot - DevTools 介绍 (http://blog.csdn.net/isea533/article/details/70495714)

静态资源处理

Spring Boot 默认的处理方式就已经足够了，默认情况下Spring Boot 使用 `WebMvcAutoConfiguration` 中配置的各种属性。

建议使用Spring Boot 默认处理方式，需要自己配置的地方可以通过配置文件修改。

但是如果你想**完全控制**Spring MVC，你可以在 `@Configuration` 注解的配置类上增加 `@EnableWebMvc`，增加该注解以后 `WebMvcAutoConfiguration` 中配置就不会生效，你需要自己来配置需要的每一项。这种情况下的配置方法建议参考 `WebMvcAutoConfiguration` 类。

本文以下内容针对Spring Boot 默认的处理方式，部分配置通过在 `application.yml` 配置文件中设置。

配置资源映射

Spring Boot 默认配置的 `/**` 映射到 `/static`（或 `/public`，`/resources`，`/META-INF/resources`），`/webjars/**` 会映射到 `classpath:/META-INF/resources/webjars/`。

注意：上面的 `/static` 等目录都是在 `classpath:` 下面。

如果你想增加如 `/mystatic/**` 映射到 `classpath:/mystatic/`，你可以让你的配置类继承 `WebMvcConfigurerAdapter`，然后重写如下方法：

```
1 @Override
2 public void addResourceHandlers(ResourceHandlerRegistry registry) {
3     registry.addResourceHandler("/mystatic/**")
4         .addResourceLocations("classpath:/mystatic/");
5 }
```

这种方式会在默认的基础上增加 `/mystatic/**` 映射到 `classpath:/mystatic/`，不会影响默认的方式，可以同时使用。

静态资源映射还有一个配置选项，为了简单这里用 `.properties` 方式书写：



isea533 (http://blog.cs...

+ 关注

(http://blog.csdn.net/isea533)

码云

原创

217

粉丝

喜欢

0

41

(https://c

utm_sour

他的最新文章

更多文章 (http://blog.csdn.net/isea533)

JMX 入门（三）

(/isea533/article/details/77600542)

MyBatis 示例之存储过程（三）

(/isea533/article/details/77592456)

MySQL 5.7 UPDATE 和 DELETE 导致的 error code [1093]

(/isea533/article/details/77531148)

JMX 入门（二）

(/isea533/article/details/77455973)

在线课程

```
1 spring.mvc.static-path-pattern=/** # Path pattern used for static resources.
```

这个配置会影响默认的 `/**`，例如修改为 `/static/**` 后，只能映射如 `/static/js/sample.js` 这样的请求（修改前是 `/js/sample.js`）。这个配置只能写一个值，不像大多数可以配置多个用逗号隔开的。

使用注意

例如有如下目录结构：

```
1  └─resources
2    │   application.yml
3    │
4    └─static
5        │   └─css
6            │       index.css
7            │
8            └─js
9                │   index.js
10               │
11               └─templates
12                   │   index.ftl
```

在 `index.ftl` 中该如何引用上面的静态资源呢？

如下写法：

```
1 <link rel="stylesheet" type="text/css" href="/css/index.css">
2 <script type="text/javascript" src="/js/index.js"></script>
```

注意：默认配置的 `/**` 映射到 `/static`（或 `/public`，`/resources`，`/META-INF/resources`）

当请求 `/css/index.css` 的时候，Spring MVC 会在 `/static/` 目录下面找到。

如果配置为 `/static/css/index.css`，那么上面配置的几个目录下面都没有 `/static` 目录，因此会找不到资源文件！

所以写静态资源位置的时候，不要带上映射的目录名（如 `/static/`，`/public/`，`/resources/`，`/META-INF/resources/`）！

使用WebJars

WebJars：<http://www.webjars.org/> (<http://www.webjars.org/>)

例如使用 `jquery`，添加依赖：

```
1 <dependency>
2   <groupId>org.webjars</groupId>
3   <artifactId>jquery</artifactId>
4   <version>1.11.3</version>
5 </dependency>
```

然后可以如下使用：

```
1 <script type="text/javascript" src="/webjars/jquery/1.11.3/jquery.js"></script>
```

你可能注意到 `href` 中的 `1.11.3` 版本号了，如果仅仅这么使用，那么当我们切换版本号的时候还要手动修改 `href`，怪麻烦的，我们可以用如下方式解决。

先在 `pom.xml` 中添加依赖：

```
1 <dependency>
2   <groupId>org.webjars</groupId>
3   <artifactId>webjars-locator</artifactId>
4 </dependency>
```

增加一个 `WebJarController`：

返回顶部

在线课程

```
1 @Controller
2 public class WebJarController {
3     private final WebJarAssetLocator assetLocator = new WebJarAssetLocator();
4
5     @ResponseBody
6     @RequestMapping("/webjarslocator/{webjar}/**")
7     public ResponseEntity locateWebjarAsset(@PathVariable String webjar, HttpServletRequest request) {
8         try {
9             String mvcPrefix = "/webjarslocator/" + webjar + "/";
10            String mvcPath = (String) request.getAttribute(HandlerMapping.PATH_WITHIN_HANDLER_MAPPING_ATTRIBUTE);
11            String fullPath = assetLocator.getFullPath(webjar, mvcPath.substring(mvcPrefix.length()));
12
13            return new ResponseEntity(new ClassPathResource(fullPath), HttpStatus.OK);
14        } catch (Exception e) {
15            return new ResponseEntity<>(HttpStatus.NOT_FOUND);
16        }
17    }
18 }
```

然后使用的时候按照如下方式：

```
1 <script type="text/javascript" src="/webjarslocator/jquery/jquery.js"></script>
```

注意：这里不需要在写版本号了，但是注意写url的时候，只是在原来url基础上去掉了版本号，其他的都不能少！

静态资源版本管理

Spring MVC 提供了静态资源版本映射的功能。

用途：当我们资源内容发生变化时，由于浏览器缓存，用户本地的静态资源还是旧的资源，为了防止这种情况导致的问题，我们可能会手动在请求url的时候加个版本号或者其他方式。

版本号如：

```
1 <script type="text/javascript" src="/js/sample.js?v=1.0.1"></script>
```

Spring MVC 提供的功能可以很容易的帮助我们解决类似问题。

Spring MVC 有两种解决方式。

注意：下面的配置方式针对 freemarker 模板方式，其他的配置方式可以参考。

资源名-md5 方式

例如：

```
1 <link rel="stylesheet" type="text/css" href="/css/index-2b371326aa93ce4b611853a309b69b29.css">
```

Spring 会自动读取资源md5，然后添加到 index.css 的名字后面，因此当资源内容发生变化的时候，文件名发生变化，就会更新本地资源。

配置方式：

在 application.properties 中做如下配置：

```
1 spring.resources.chain.strategy.content.enabled=true
2 spring.resources.chain.strategy.content.paths=/**
```

这样配置后，所有 /** 请求的静态资源都会被处理为上面例子的样子。

到这儿还没完，我们在写资源url的时候还要特殊处理。

首先增加如下配置：

返回顶部

在线课程

```
1 @ControllerAdvice
2 public class ControllerConfig {
3
4     @Autowired
5     ResourceUrlProvider resourceUrlProvider;
6
7     @ModelAttribute("urls")
8     public ResourceUrlProvider urls() {
9         return this.resourceUrlProvider;
10    }
11
12 }
```

然后在页面写的时候用下面的写法：

```
1 <link rel="stylesheet" type="text/css" href="${urls.getForLookupPath('/css/index.css')}">
```

使用 `urls.getForLookupPath('/css/index.css')` 来得到处理后的资源名。

版本号 方式

在 `application.properties` 中做如下配置：

```
1 spring.resources.chain.strategy.fixed.enabled=true
2 spring.resources.chain.strategy.fixed.paths=/js/**,/v1.0.0/**
3 spring.resources.chain.strategy.fixed.version=v1.0.0
```

这里配置需要特别注意，将 `version` 的值配置在 `paths` 中。原因我们在讲Spring MVC 处理逻辑的时候说。

在页面写的时候，写法如下：

```
1 <script type="text/javascript" src="${urls.getForLookupPath('/js/index.js')}"></script>
```

注意，这里仍然使用了 `urls.getForLookupPath`，`urls` 配置方式见上一种方式。

在请求的实际页面中，会显示为：

```
1 <script type="text/javascript" src="/v1.0.0/js/index.js"></script>
```

可以看到这里的地址是 `/v1.0.0/js/index.js`。

静态资源版本管理 处理过程

在Freemarker模板首先会调用 `urls.getForLookupPath` 方法，返回一个 `/v1.0.0/js/index.js` 或 `/css/index-2b371326aa93ce4b611853a309b69b29.css`。

这时页面上的内容就是处理后的资源地址。

这之后浏览器发起请求。

这里分开说。

第一种md5方式

请求 `/css/index-2b371326aa93ce4b611853a309b69b29.css`，我们md5配置的 `paths=/**`，所以Spring MVC 会尝试url中是否包含 `-`，如果包含会去掉后面这部分，然后去映射的目录（如 `/static/`）查找 `/css/index.css` 文件，如果能找到就返回。

第二种版本方式

请求 `/v1.0.0/js/index.js`。

如果我们 `paths` 中没有配置 `/v1.0.0`，那么上面这个请求地址就不会按版本方式来处理，因此会找不到上面的资源。

如果配置了 `/v1.0.0`，Spring 就会将 `/v1.0.0` 去掉再去找 `/js/index.js`，最终会在 `/static/` 下面找到。

返回顶部

本文参考

1.

<http://docs.spring.io/spring-boot/docs/current/reference/html/boot-features-developing-web-applications.html> (<http://docs.spring.io/spring-boot/docs/current/reference/html/boot-features-developing-web-applications.html>)
2.

<http://www.webjars.org/documentation> (<http://www.webjars.org/documentation>)
3.

<http://www.mscharhag.com/spring/resource-versioning-with-spring-mvc> (<http://www.mscharhag.com/spring/resource-versioning-with-spring-mvc>)
4.

<https://spring.io/blog/2014/07/24/spring-framework-4-1-handling-static-web-resources> (<https://spring.io/blog/2014/07/24/spring-framework-4-1-handling-static-web-resources>)
- 如果你使用的JSP或者其他模板，你可以参考上面几个链接的内容。

最后

以上是Spring Boot 静态资源处理的内容，有些不全面的地方或者读者有更多疑问，可以查看Spring Boot完整文档或本文参考的内容。

关于Spring Boot更多的内容可以继续关注本博客。

Spring Boot 系列

由于我博客Spring Boot 系列文章还不够多，所以暂时不打算创建专栏，如果再多几篇我就建专栏。

1.

[Spring Boot 入门](http://blog.csdn.net/isea533/article/details/50278205) (<http://blog.csdn.net/isea533/article/details/50278205>)
2.

[Spring Boot 属性配置和使用](http://blog.csdn.net/isea533/article/details/50281151) (<http://blog.csdn.net/isea533/article/details/50281151>)
3.

[Spring Boot 集成MyBatis](http://blog.csdn.net/isea533/article/details/50359390) (<http://blog.csdn.net/isea533/article/details/50359390>)
4.

[Spring Boot 静态资源处理](http://blog.csdn.net/isea533/article/details/50412212) (<http://blog.csdn.net/isea533/article/details/50412212>)

版权声明：版权归博主所有，转载请带上本文链接！联系方式：abel533@gmail.com

▲ 举报

标签：[spring](http://so.csdn.net/so/search/s.do?q=spring&t=blog) (<http://so.csdn.net/so/search/s.do?q=spring&t=blog>) /

[spring mvc](http://so.csdn.net/so/search/s.do?q=spring+mvc&t=blog) (<http://so.csdn.net/so/search/s.do?q=spring+mvc&t=blog>) /

[springboot](http://so.csdn.net/so/search/s.do?q=springboot&t=blog) (<http://so.csdn.net/so/search/s.do?q=springboot&t=blog>) /

[静态资源](http://so.csdn.net/so/search/s.do?q=静态资源&t=blog) (<http://so.csdn.net/so/search/s.do?q=静态资源&t=blog>) /

本文已收录于以下专栏：[Spring Boot 入门](http://blog.csdn.net/column/details/15358.html) (<http://blog.csdn.net/column/details/15358.html>)



发表评论

(http://my.csdn.net/qq_36596145)



sun1021873926 ([/sun1021873926](#))

2017-08-27 09:40

5楼

([/sun1021873926](#))

回复



qq_24652087 ([/qq_24652087](#))

2017-07-31 11:09

4楼

([/qq_24652087](#))

回复

成jar后,文件上传到指定目录，然后配置拦截器访问外部路径，不通。。。请指教

查看 7 条热评 ▾

在线课程



【免费】深入理解Docker内部原理及网络配置

http://edu.csdn.net/huayiCourse/detail/563?utm_source=blog9



SDCC 2017之区块链技术与实战线上峰会


http://edu.csdn.net/huayiCourse/series_detail/66?utm_source=blog9

相关文章推荐

在线课程


Spring Boot 静态资源处理 (/catoop/article/details/50501706)

Spring Boot 默认为我们提供了静态资源处理，使用 WebMvcAutoConfiguration 中的配置各种属性。建议大家使用Spring Boot的默认配置方式，如果需要特殊处理的再通过...

 catoop (http://blog.csdn.net/catoop) 2016-01-12 09:23 57068

SpringBoot加载静态资源 (/u010054969/article/details/62230350)

Spring Boot从classpath下一个叫/static (/public , /resources或/META-INF/resources) 的文件夹或从ServletContext根目录提供静态内...

 u010054969 (http://blog.csdn.net/u010054969) 2017-03-15 16:58 760




精选：深入理解 Docker 内部原理及网络配置 (http://edu.csdn.net/huiyiCourse/detail/563?utm_source=blog10)

网络绝对是任何系统的核心，对于容器而言也是如此。Docker 作为目前最火的轻量级容器技术，有很多令人称道的功能，如 Docker 的镜像管理。然而，Docker的网络一直以来都比较薄弱，所以我们有必要深入了解Docker的网络知识，以满足更高的网络需求。


Spring Boot 入门 (/isea533/article/details/50278205)

Spring Boot入门Spring Boot是Spring社区较新的一个项目。该项目的目的是帮助开发者更容易的创建基于Spring的应用程序和服务，让更多的人更快地对Spring进行入门体验，让...

 isea533 (http://blog.csdn.net/isea533) 2015-12-27 15:41 151757


java 枚举——java中枚举的运用和使用场景 (/pangesange/article/details/54573382)

转自：http://blog.csdn.net/yehui928186846/article/details/51426415 基本特性：1，enum关键字 枚举enum是同class，int...

 pangesange (http://blog.csdn.net/pangesange) 2017-01-16 13:40 857


Spring Boot 属性配置和使用 (/isea533/article/details/50281151)

Spring Boot 属性配置和使用Spring Boot 允许通过外部配置让你在不同的环境使用同一应用程序的代码，简单说就是通过配置文件来注入属性或者修改默认的配置。Spring Boot 入...

 isea533 (http://blog.csdn.net/isea533) 2015-12-27 15:27 150322

Spring Boot 集成MyBatis (/isea533/article/details/50359390)

Spring Boot 集成MyBatis在配置MyBatis前，我们先配置一个druid数据源。Spring Boot 集成druid有很多个配置选项，使用Spring Boot 的Con...

 isea533 (http://blog.csdn.net/isea533) 2015-12-27 15:29 188116

SpringBoot非官方教程 | 第二十五篇：2小时学会springboot (/forezp/article/details/61472783)



一.什么是spring boot Takes an opinionated view of building production-ready Spring applications. Sprin...

 forezp (http://blog.csdn.net/forezp) 2017-03-12 00:23 13252

返回顶部

java枚举常用场景小结 (/cndmss/article/details/51441617)

在java编程过程中，我们通常需要定义一些固定数量的常量，在jdk1.5以前，通常的做法是定义一个静态常量类，但自jdk1.5后，java引入了枚举（关键字enum，全称为 enumeration，值...

 cndmss (<http://blog.csdn.net/cndmss>) 2016-05-18 10:10  2614



史上最简单的 SpringCloud 教程 | 第一篇：服务的注册与发现（Eureka） (/forezp/article/details/69696915)

spring cloud 为开发人员提供了快速构建分布式系统的一些工具，包括配置管理、服务发现、断路器、路由、微代理、事件总线、全局锁、决策竞选、分布式会话等等。它运行环境简单，可以在开发人员的电脑上...

 forezp (<http://blog.csdn.net/forezp>) 2017-04-08 18:16  49357



Spring Boot 静态资源处理 (/zhao1949/article/details/55250551)

<http://www.2cto.com/kf/201601/485530.html> spring-boot配置外部静态资源的方法 <http://www.cnblogs.com/feika/p/45...>

 zhao1949 (<http://blog.csdn.net/zhao1949>) 2017-02-16 08:20  432



spring boot 静态资源处理 (/yingxiake/article/details/51295551)

spring boot 秉承约定优于配置，spring boot在静态资源的处理上就已经默认做了处理。1.默认资源映射映射“/**” 的路径到 /static （或/public、/resources、...

 yingxiake (<http://blog.csdn.net/yingxiake>) 2016-05-02 11:52  4433

Spring Boot 静态资源处理 (/isea533/article/details/50412212)

Spring Boot 静态资源处理Spring Boot 默认的处理方式就已经足够了，默认情况下Spring Boot 使用WebMvcAutoConfiguration中配置的各种属性。建议使用S...

 isea533 (<http://blog.csdn.net/isea533>) 2015-12-27 15:38  69718

Spring Boot 静态资源处理 (/zlfprogram/article/details/75309036)

静态资源处理spring Boot 默认的处理方式就已经足够了，默认情况下Spring Boot 使用WebMvcAutoConfiguration中配置的各种属性。建议使用Spring Boot 默...

 zlfprogram (<http://blog.csdn.net/zlfprogram>) 2017-07-18 16:26  1818



Spring Boot 静态资源处理 (/lsy0903/article/details/52910651)

Spring Boot 默认为我们提供了静态资源处理，使用 WebMvcAutoConfiguration 中的配置各种属性。建议大家使用Spring Boot的默认配置方式，如果需要特殊处理的再通过...

 lsy0903 (<http://blog.csdn.net/lsy0903>) 2016-10-24 12:52  216

Spring Boot 静态资源处理 (/u011955252/article/details/53173293)



spring Boot 默认为我们提供了静态资源处理，使用 WebMvcAutoConfiguration 中的配置各种属性。 建议大家使用Spring Boot的默认配置方式，如果需要特殊处理...

 u011955252 (<http://blog.csdn.net/u011955252>) 2016-11-15 16:57  797

Spring静态资源处理 (/qq_16465949/article/details/51907503)

在线课程

针对静态文件处理的方法总结 动静分离一直是比较好的提高网站响应速度的方案。从网上搜索的资料来看有三种有关Spring与容器的方式。另外还有一些细节注意，否则会使controller失效。 第一种 ...

 qq_16465949 (http://blog.csdn.net/qq_16465949) 2016-07-14 11:11  732



Spring MVC 处理静态资源 (/mdreamlove/article/details/62927261)

优雅的REST风格的资源URL不希望带 .html 或 .do 等后缀 若将 DispatcherServlet 请求映射配置为 / ，则Spring MVC将捕获 WEB 容器的所有请求，包括静态资...

 MDreamlove (<http://blog.csdn.net/MDreamlove>) 2017-03-17 22:43  264



Spring MVC静态资源处理 (/aosica321/article/details/54580297)

来源： <http://www.cnblogs.com/fangqi/archive/2012/10/28/2743108.html> 优雅REST风格的资源URL不希望带 .html 或 .do...

 aosica321 (<http://blog.csdn.net/aosica321>) 2017-01-17 09:50  53

Spring MVC静态资源处理 (/u010414666/article/details/51282796)

优雅REST风格的资源URL不希望带 .html 或 .do 等后缀.由于早期的Spring MVC不能很好地处理静态资源，所以在web.xml中配置DispatcherServlet的请求映射，往往...

 u010414666 (<http://blog.csdn.net/u010414666>) 2016-04-29 16:31  349

在线课程