SpringBoot Thymeleaf 模板引擎渲染视图

作者：Javen
原文链接：https://ld246.com/article/1497247102825
来源网站：链滴

静态资源访问

在我们开发Web应用的时候，需要引用大量的js、css、图片等静态资源。

默认配置

Spring Boot默认提供静态资源目录位置需置于classpath下，目录名需符合如下规则：

/static

/public

/resources

/META-INF/resources

举例：我们可以在src/main/resources/目录下创建static，在该位置放置一个图片文件。启动程序后尝试访问http://localhost:8080/D.jpg。如能显示图片，配置成功。

渲染Web页面

在之前的示例中，我们都是通过@RestController来处理请求，所以返回的内容为json对象。那么如需要渲染html页面的时候，要如何实现呢？

模板引擎

在动态HTML实现上Spring Boot依然可以完美胜任，并且提供了多种模板引擎的默认配置支持，所在推荐的模板引擎下，我们可以很快的上手开发动态网站。

Spring Boot提供了默认配置的模板引擎主要有以下几种：

Thymeleaf

FreeMarker

Velocity

Groovy

Mustache

Spring Boot建议使用这些模板引擎，避免使用JSP，若一定要使用JSP将无法实现Spring Boot的多特性，具体可见后文：支持JSP的配置

当你使用上述模板引擎中的任何一个，它们默认的模板配置路径为：src/main/resources/templates 当然也可以修改这个路径，具体如何修改，可在后续各模板引擎的配置属性中查询并修改。

Thymeleaf

Thymeleaf是一个XML/XHTML/HTML5模板引擎，可用于Web与非Web环境中的应用开发。它是一开源的Java库，基于Apache License 2.0许可，由Daniel Fernández创建，该作者还是Java加密库Jaspt的作者。

Thymeleaf提供了一个用于整合Spring MVC的可选模块，在应用开发中，你可以使用Thymeleaf来全代替JSP或其他模板引擎，如Velocity、FreeMarker等。Thymeleaf的主要目标在于提供一种可被

览器正确显示的、格式良好的模板创建方式，因此也可以用作静态建模。你可以使用它创建经过验证XML与HTML模板。相对于编写逻辑或代码，开发者只需将标签属性添加到模板中即可。接下来，这标签属性就会在DOM（文档对象模型）上执行预先制定好的逻辑。

示例模板：

```
<table>
  <thead>
    <tr>
      <th th:text="#{msgs.headers.name}">Name</td>
      <th th:text="#{msgs.headers.price}">Price</td>
    </tr>
  </thead>
  <tbody>
    <tr th:each="prod : ${allProducts}">
      <td th:text="${prod.name}">Oranges</td>
      <td th:text="${#numbers.formatDecimal(prod.price,1,2)}">0.99</td>
    </tr>
  </tbody>
</table>
```

可以看到Thymeleaf主要以属性的方式加入到html标签中，浏览器在解析html时，当检查到没有的属时候会忽略，所以Thymeleaf的模板可以通过浏览器直接打开展现，这样非常有利于前后端的分离。

在Spring Boot中使用Thymeleaf，只需要引入下面依赖，并在默认的模板路径src/main/resources/tmplates下编写模板文件即可完成。

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

在完成配置之后，举一个简单的例子，在快速入门工程的基础上，举一个简单的示例来通过Thymelea渲染一个页面。

```
@Controller
public class HelloController {
    @RequestMapping("/")
    public String index(ModelMap map) {
        // 加入一个属性，用来在模板中读取
        map.addAttribute("host", "http://blog.didispace.com");
        // return模板文件的名称，对应src/main/resources/templates/index.html
        return "index";
    }
}
```

```
<!DOCTYPE html>
<html>
<head lang="en">
    <meta charset="UTF-8" />
```

```
    <title></title>
</head>
<body>
<h1 th:text="${host}">Hello World</h1>
</body>
</html>
```

如上页面，直接打开html页面展现Hello World，但是启动程序后，访问http://localhost:8080/，则展示Controller中host的值：http://blog.didispace.com，做到了不破坏HTML自身内容的数据逻辑离。

更多Thymeleaf的页面语法，还请访问Thymeleaf的官方文档查询使用。

Thymeleaf的默认参数配置

如有需要修改默认配置的时候，只需复制下面要修改的属性到application.properties中，并修改成要的值，如修改模板文件的扩展名，修改默认的模板路径等。

```
# Enable template caching.
spring.thymeleaf.cache=true
# Check that the templates location exists.
spring.thymeleaf.check-template-location=true
# Content-Type value.
spring.thymeleaf.content-type=text/html
# Enable MVC Thymeleaf view resolution.
spring.thymeleaf.enabled=true
# Template encoding.
spring.thymeleaf.encoding=UTF-8
# Comma-separated list of view names that should be excluded from resolution.
spring.thymeleaf.excluded-view-names=
# Template mode to be applied to templates. See also StandardTemplateModeHandlers.
spring.thymeleaf.mode=HTML5
# Prefix that gets prepended to view names when building a URL.
spring.thymeleaf.prefix=classpath:/templates/
# Suffix that gets appended to view names when building a URL.
spring.thymeleaf.suffix=.html  spring.thymeleaf.template-resolver-order= # Or
```