链滴

SpringMVC 后缀名为.html 获取 json， 报 406 (Not Acceptable)

作者：meikaiyipian

原文链接：https://ld246.com/article/1514002399162

来源网站：链滴

访问时浏览器console 报 the server responded with a status of 406 (Not Acceptable)。

http 406 响应码：HTTP 406错误是HTTP协议状态码的一种，表示无法使用请求的内容特性来响应求的网页。一般是指客户端浏览器不接受所请求页面的 MIME 类型。

SpringMVC DispatcherServlet 配置：

```
<servlet>
    <servlet-name>springmvc</servlet-name>
    <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
     <init-param>
        <param-name>contextConfigLocation</param-name>
        <param-value>classpath:spring-servlet.xml</param-value>
     </init-param>
     <load-on-startup>1</load-on-startup>
</servlet>

<servlet-mapping>
    <servlet-name>springmvc</servlet-name>
    <url-pattern>*.html</url-pattern>
</servlet-mapping>
```

Controller 层代码：

```
@ResponseBody
@RequestMapping("listTag")
public List<Tag> listTag(HttpServletRequest request) {
    final Tag tagParam = new Tag();
    tagParam.setUserId(getSessionUserId(request));

    return tagService.listByUser(tagParam);
}
```

以上 listTag 方法响应的应是一个 json 串，但在浏览器访问的时候，查看响应头却是 text/html ，响与预期不符，浏览器不接受这样的类型，所以就报 406 错误。

Chrome F12 Network Headers:

```
General:
Request URL:http://localhost:8080/tag/listTag.html
Request Method:GET
Status Code:406 Not Acceptable
Remote Address:127.0.0.1:8080
Referrer Policy:no-referrer-when-downgrade

Response Headers:
Content-Language:en
Content-Length:1067
Content-Type:text/html;charset=utf-8
Date:Tue, 05 Dec 2017 16:02:20 GMT
Server:Apache-Coyote/1.1

Request Headers:
Accept:text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q
```

```
0.8
Accept-Encoding:gzip, deflate, br
Accept-Language:zh-CN,zh;q=0.9,en-US;q=0.8,en;q=0.7
Cache-Control:max-age=0
Connection:keep-alive
Cookie:_ga=GA1.1.1540456839.1503116139; Idea-a616d67b=f46cb096-4fbe-4162-949e-c3ad
a9fb357; JSESSIONID=A4B212DEFD86FB99934B7B510495180D
DNT:1
Host:localhost:8080
Upgrade-Insecure-Requests:1
User-Agent:Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gec
o) Chrome/62.0.3202.94 Safari/537.36
```

出现该问题的主要原因是因为 SpringMVC 默认根据 url 后缀名来判断应以什么样的方式进行响应。

SpringMVC 有专门一个 org.springframework.http.MediaType 类定义了这些 MIME 常量：

```
public class MediaType extends MimeType implements Serializable {
    private static final long serialVersionUID = 2069937152339670231L;
    public static final MediaType ALL = valueOf("*/*");
    public static final String ALL_VALUE = "*/*";
    public static final MediaType APPLICATION_ATOM_XML = valueOf("application/atom+xml")

    public static final String APPLICATION_ATOM_XML_VALUE = "application/atom+xml";
    public static final MediaType APPLICATION_FORM_URLENCODED = valueOf("application/x
www-form-urlencoded");
    public static final String APPLICATION_FORM_URLENCODED_VALUE = "application/x-www-
orm-urlencoded";
    public static final MediaType APPLICATION_JSON = valueOf("application/json");
    public static final String APPLICATION_JSON_VALUE = "application/json";
    public static final MediaType APPLICATION_JSON_UTF8 = valueOf("application/json;charse
=UTF-8");
    public static final String APPLICATION_JSON_UTF8_VALUE = "application/json;charset=UTF
8";
    public static final MediaType APPLICATION_OCTET_STREAM = valueOf("application/octet-s
ream");
    public static final String APPLICATION_OCTET_STREAM_VALUE = "application/octet-stream"

    public static final MediaType APPLICATION_PDF = valueOf("application/pdf");
    public static final String APPLICATION_PDF_VALUE = "application/pdf";
    public static final MediaType APPLICATION_RSS_XML = valueOf("application/rss+xml");
    public static final String APPLICATION_RSS_XML_VALUE = "application/rss+xml";
    public static final MediaType APPLICATION_XHTML_XML = valueOf("application/xhtml+xm
");
    public static final String APPLICATION_XHTML_XML_VALUE = "application/xhtml+xml";
    public static final MediaType APPLICATION_XML = valueOf("application/xml");
    public static final String APPLICATION_XML_VALUE = "application/xml";
    public static final MediaType IMAGE_GIF = valueOf("image/gif");
    public static final String IMAGE_GIF_VALUE = "image/gif";
    public static final MediaType IMAGE_JPEG = valueOf("image/jpeg");
    public static final String IMAGE_JPEG_VALUE = "image/jpeg";
    public static final MediaType IMAGE_PNG = valueOf("image/png");
    public static final String IMAGE_PNG_VALUE = "image/png";
    public static final MediaType MULTIPART_FORM_DATA = valueOf("multipart/form-data");
```

```
    public static final String MULTIPART_FORM_DATA_VALUE = "multipart/form-data";
    public static final MediaType TEXT_EVENT_STREAM = valueOf("text/event-stream");
    public static final String TEXT_EVENT_STREAM_VALUE = "text/event-stream";
    public static final MediaType TEXT_HTML = valueOf("text/html");
    public static final String TEXT_HTML_VALUE = "text/html";
    public static final MediaType TEXT_MARKDOWN = valueOf("text/markdown");
    public static final String TEXT_MARKDOWN_VALUE = "text/markdown";
    public static final MediaType TEXT_PLAIN = valueOf("text/plain");
    public static final String TEXT_PLAIN_VALUE = "text/plain";
    public static final MediaType TEXT_XML = valueOf("text/xml");
    public static final String TEXT_XML_VALUE = "text/xml";
    private static final String PARAM_QUALITY_FACTOR = "q";
...
}
```

因此，若是请求 url 的后缀名特殊时，需要进行一定的响应配置。

SpringMVC 响应配置一般我们是直接配置：

```
<mvc:annotation-driven/>
```

SpringMVC 有一个专门用来配置多视图请求格式的 FactoryBean 对象：org.springframework.web.accept.ContentNegotiationManagerFactoryBean，可以用此对象配置以 .html 为后缀的请求响应式。

修改上面的配置为：

```
<mvc:annotation-driven content-negotiation-manager="contentNegotiationManager" />

<!-- 以.html为后缀名访问，默认响应的数据类型是 text/html， 将其响应类型修改为 application/json -->
<bean id="contentNegotiationManager" class="org.springframework.web.accept.ContentNegotiationManagerFactoryBean">
    <property name="mediaTypes">
        <map>
            <entry key="html" value="application/json;charset=UTF-8"/>
        </map>
    </property>
</bean>
```

另外，SpringMVC Controller 方法若使用 @ResponseBody 返回 json 格式数据，需要依赖于 jackson-core、jackson-databind、jackson-annotations jar，这些 jar 可以到 maven 仓库中下载。

也许会有人问，为什么一定要以 .html 作为后缀名呢？改成其他普通的后缀不就没这个问题了吗？

这是因为想要增强搜索引擎的友好性，众所周知，搜索引擎更青睐于静态页面，但很多时候为了保证面的实时性，大多用的是动态页面（jsp asp php）。若想要两者均衡，就可以使用伪静态技术。