

# SpringMVC的消息转换器

参考视频或文章：

- ↗[https://blog.csdn.net/m0\\_50742275/article/details/144899648?fromshare=blogdetail&sharetype=blogdetail&sharerId=144899648&shareref=PC&sharesource=2401\\_83600210&sharefrom=from\\_link](https://blog.csdn.net/m0_50742275/article/details/144899648?fromshare=blogdetail&sharetype=blogdetail&sharerId=144899648&shareref=PC&sharesource=2401_83600210&sharefrom=from_link)

## 一、技术介绍

### 1.概述

- SpringMVC的消息转换器(Message Converter)是Spring框架中用于处理HTTP请求体或响应体与Java对象之间转换的组件。
- 让开发人员可以轻松地将HTTP请求的数据映射到方法参数，并将返回的对象转换为HTTP响应。

### 2.工作原理

当一个HTTP请求到达SpringMVC应用程序时，Spring框架会根据该请求的 `Content-Type` (内容类型) 和 `Accept` (接受类型) 来选择合适的消息转换器。

例如：客户端发送了一个JSON格式的POST请求，那么SpringMVC会选择 `MappingJackson2HttpMessageConverter` 这一消息转换器来将请求体反序列化为Java对象；当方法返回一个Java对象并需要将其发送给客户端时，SpringMVC仍会使用这一消息转换器来将这个Java对象序列化为JSON格式的响应体。

### 3. 常见的内置消息转换器

消息转换器	功能	依赖/适用场景
MappingJackson2HttpMessageConverter	支持JSON格式的HTTP消息序列化与反序列化	依赖Jackson库
MappingJackson2XmlHttpMessageConverter	支持XML格式的HTTP消息序列化与反序列化	依赖Jackson库
StringHttpMessageConverter	处理纯文本字符串类型的HTTP消息	无特殊依赖，适用于文本传输场景
FormHttpMessageConverter	处理表单数据格式的HTTP消息	支持 <code>application/x-www-form-urlencoded</code> 和 <code>multipart/form-data</code> 类型，可用于标准表单提交和文件上传
ByteArrayHttpMessageConverter	处理二进制数据格式的HTTP消息	适用于图片传输、文件下载等二进制数据场景
Jaxb2RootElementHttpMessageConverter	实现XML数据的序列化与反序列化	基于JAXB API
SourceHttpMessageConverter	处理基于 <code>javax.xml.transform.Source</code> 的XML消息	适用于XML源数据的传输场景
ResourceHttpMessageConverter	处理资源文件类型的HTTP消息	适用于文件下载等资源访问场景

## 二、项目应用

### 1.项目需求

初步开发完员工管理模块的员工分页查询功能后，测试发现如下问题：

员工姓名	账号	手机号	账号状态	最后操作时间	操作
王五	wangwu	13312345678	启用	202292895820	<a href="#">修改</a>   <a href="#">禁用</a>
李四	lisi	13212345678	启用	2022927152115	<a href="#">修改</a>   <a href="#">禁用</a>
张三	zhangsan	13812344321	启用	202292715184	<a href="#">修改</a>   <a href="#">禁用</a>
管理员	admin	13812312312	启用	2022920185540	<a href="#">修改</a>   <a href="#">禁用</a>

可以看到，最后操作时间的格式并不符合预期，这是因为后端响应给前端的日期时间数据没有进行格式化处理。

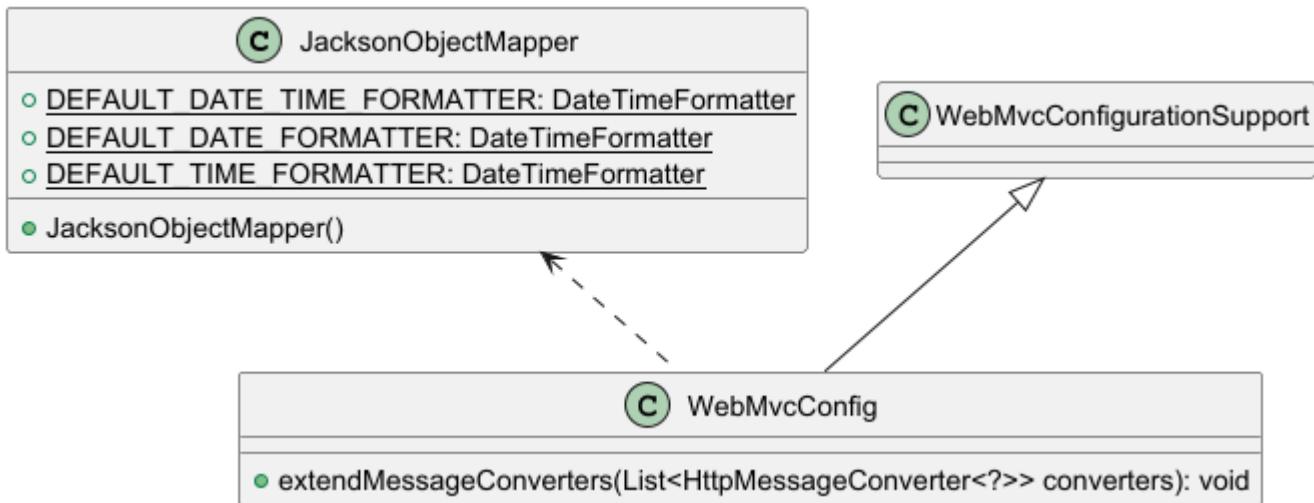
解决方法有以下2种：

- 单个处理：**在相关属性上加上`@JsonFormat(pattern="")`注解，对日期时间进行格式化处理；
- 统一处理：**在`WebMvcConfig`中扩展SpringMVC的消息转换器，统一对日期时间类型的数据进行格式化处理。

### 2.扩展SpringMVC的消息转换器

涉及到的文件如下：

```
1 sky-common:  
2     json: JacksonObjectMapper  
3  
4 sky-server:  
5     config: WebMvcConfig
```



## 2.1 创建对象转换器 JacksonObjectMapper

```

1 /**
2  * 对象转换器：基于jackson将Java对象转为json，或者将json转为Java对象
3  * 将JSON解析为Java对象的过程称为 [从JSON反序列化Java对象]
4  * 从Java对象生成JSON的过程称为 [序列化Java对象到JSON]
5 */
6 public class JacksonObjectMapper extends ObjectMapper {
7
8     public static final DateTimeFormatter DEFAULT_DATE_TIME_FORMATTER =
9         DateTimeFormatter.ofPattern("yyyy-MM-dd HH:mm:ss");
10    public static final DateTimeFormatter DEFAULT_DATE_FORMATTER =
11        DateTimeFormatter.ofPattern("yyyy-MM-dd");
12    public static final DateTimeFormatter DEFAULT_TIME_FORMATTER =
13        DateTimeFormatter.ofPattern("HH:mm:ss");
14
15    public JacksonObjectMapper() {
16        super();
17        // 收到未知属性时不报异常
18        this.configure(FAIL_ON_UNKNOWN_PROPERTIES, false);
19
20        // 反序列化时，属性不存在的兼容处理
21
22        this.getDeserializationConfig().withoutFeatures(DeserializationFeature
23            .FAIL_ON_UNKNOWN_PROPERTIES);
24
25        SimpleModule simpleModule = new SimpleModule()
26            .addDeserializer(LocalDateTime.class, new
27                LocalDateTimeDeserializer(DEFAULT_DATE_TIME_FORMATTER))
  
```

```

22         .addDeserializer(LocalDate.class, new
23             LocalDateDeserializer(DEFAULT_DATE_FORMATTER))
24         .addDeserializer(LocalTime.class, new
25             LocalTimeDeserializer(DEFAULT_TIME_FORMATTER))
26         .addSerializer(LocalDateTime.class, new
27             LocalDateTimeSerializer(DEFAULT_DATE_TIME_FORMATTER))
28         .addSerializer(LocalDate.class, new
29             LocalDateSerializer(DEFAULT_DATE_FORMATTER))
30         .addSerializer(LocalTime.class, new
31             LocalTimeSerializer(DEFAULT_TIME_FORMATTER));
32
33     // 注册功能模块：例如，可以添加自定义序列化器和反序列化器
34     this.registerModule(simpleModule);
35 }
36 }
```

## 2.2 在 `WebMvcConfig` 中扩展 SpringMVC 的消息转换器

```

1 /**
2  * 配置类，注册web层相关组件
3 */
4 @Configuration
5 public class WebMvcConfig extends WebMvcConfigurationSupport {
6
7     // 设置静态资源映射
8     @Override
9     protected void addResourceHandlers(ResourceHandlerRegistry
10        registry) {
11
12         registry.addResourceHandler("/doc.html").addResourceLocations("classpa
13         th:/META-INF/resources/");
14
15         registry.addResourceHandler("/webjars/**").addResourceLocations("class
16         path:/META-INF/resources/webjars/");
17     }
18
19
20     // 扩展SpringMVC消息转换器，统一对日期时间类型进行格式化处理
21     @Override
22     protected void extendMessageConverters(List<HttpMessageConverter<?
23         >> converters) {
24
25 }
```

```
18 // 创建一个消息转换器对象
19 MappingJackson2HttpMessageConverter converter = new
MappingJackson2HttpMessageConverter();
20 // 设置对象转换器，可以将Java对象序列化为json字符串
21 converter.setObjectMapper(new JacksonObjectMapper());
22 // 将我们设置的对象转换器放入SpringMVC的消息转换器中的第一位置，最优先
使用
23 converters.add(0, converter);
24 }
25 }
```