使用 Spring Boot 配置日志

已翻译 100%

oschina 投递于 2017/09/09 08:16 (共 19 段, 翻译完成于 09-19) 阅读 2463 收藏 130

英文原文：Configuring Logback With Spring Boot

参与翻译 (6人) : 亚林瓜子, Tocy, vampire88, 回忆悲哀烦恼, 风马少年, aaronday

当您使用 Spring Boot 启动时，因为包含了 spring-boot-starter-logging ，让 Logback 为 Spring Boot 提供开箱即用的日志回溯——即提供日志记录，而不需要任何配置，并可以根据需求进行更改。

提供自己的配置有两种方法：如果只需要简单的更改，可以将它们添加到属性文件中，例如 application。如果是要更改属性或更复杂的需求，可以使用 XML 或 Groovy 来指定设置。

在本教程中，我们将专注于使用 XML 定义自定义日志记录配置，并查看一些基本操作，以及简要介绍使用属性文件来指定 Spring Boot 提供的标准配置以及关于它的一些简单更改。

亚林瓜子

翻译于 2017/09/09 09:42

前面我提到使用 spring-boot-starter 这个 dependency 来引入 spring-boot-start-logging ， 参照如下代码。

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

```
<dependencies>
    <dependency>
        <groupId>ch.qos.logback</groupId>
        <artifactId>logback-classic</artifactId>
    </dependency>
    <dependency>
        <groupId>org.slf4j</groupId>
        <artifactId>jul-to-slf4j</artifactId>
    </dependency>
    <dependency>
        <groupId>org.slf4j</groupId>
        <artifactId>log4j-over-slf4j</artifactId>
    </dependency>
</dependencies>
```

logback-classic 包含 dependency”logback-core“，且它们之间包含着手时所需要的一切。Spring Boot logging guide 提到基于 jcl-over-slf4j 的 dependency ，但是在 2.0.0.M3 版本中使用 spring-boot-starter-parent 时找不到它，所以我猜想有人在某处移除了这个 dependency 。 但是如果您正在使用当前 1.5.6.RELEASE 版本，它确实是存在的。

回忆悲哀烦恼

翻译于 2017/09/09 10:21

顶 0

在开始具体配置 Logback 各个配置项之前，先简单了解一下如何在一个类里面记录日志信息。

```
@Service public class MyServiceImpl implements MyService {
    private static final Logger LOGGER = LoggerFactory.getLogger(MyServiceImpl.class);
    @Override public void doStuff(final String value) {
        LOGGER.trace("doStuff needed more information - {}", value);
        LOGGER.debug("doStuff needed to debug - {}", value);
        LOGGER.info("doStuff took input - {}", value);
        LOGGER.warn("doStuff needed to warn - {}", value);
        LOGGER.error("doStuff encountered an error with value - {}", value);
    }
}
```

下面用一个相对简单的例子来熟悉一下 Logback 配置文件的配置项。Logback 会寻找项目中特定文件来配置 Logback 日志记录的设置，一般这个文件我们会命名为 logback.xml。

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender">
        <encoder>
            <pattern>%d{dd-MM-yyyy HH:mm:ss.SSS} %magenta([%thread]) %highlight(%-5level) %logger{36}.
        </encoder>
    </appender>
    <root level="info">
        <appender-ref ref="STDOUT" />
    </root>
</configuration>
```

风

风马少年

翻译于 2017/09/10
16:56

它将创建一个 ConsoleAppender 类的 appender ，它会将日志消息输出到控制台，就像 System.out.print 一样。设置日志消息将遵循的模式，其中提供一些符号，这些符号根据已发送到记录器的消息来替换生成的值。示例中已经包含了一些符号。以下是对每项符号的解释：

- %d - 以 SimpleDateFormat 允许的格式输出日志消息发生的时间。
- %thread - 输出日志消息发生的线程的名称。
- $ -5level - 输出日志消息的日志级别。
- %logger {36} - 输出日志消息发生的包+类名。括号中的数字表示包+类名的最大长度。如果输出长于指定的长度，则从根包中开始，每个包的第一个字符的子串将从根包开始直到输出低于最大长度。类名永远不会减少。转换文字文档中可以找到一个很好的案例。
- %M - 输出日志消息发生的方法的名称（使用起来很慢，不推荐，除非你不担心性能，或者方法名称对你尤其重要）。
- %msg - 输出实际的日志消息。
- %n - 换行。

- highlight（）- 根据记录级别（例如ERROR - red）设置指定中包含的输出的颜色高亮。

亚林瓜子

翻译于 2017/09/12 08:47

顶 1

之前创建的 appender 是在根 logger 中引用的。在上述示例中，日志记录级别已设置为 INFO（可以使用小写或大写）。使系统仅输出日志级别在 INFO 或更高（INFO，WARN，ERROR）上定义的消息。

Logback 中的可用日志记录级别为：

- OFF (不输出日志)
- ERROR
- WARN
- INFO
- DEBUG
- TRACE

回到上面已经展示的使用日志级别为 INFO 的代码段，其中只有级别在 INFO 或更高（ WARN 和 ERROR ）的消息才会输出到日志中。因此，如果我们调用 MyService.doStuff(“value”)，它将输出以下内容（与 Spring 相关的日志均已从当前和后续示例中删除）。

```
8-08-2017 13:32:18.549 [main] INFO  com.lankydan.service.MyServiceImpl.doStuff - doStuff took input -
28-08-2017 13:32:18.549 [main] WARN  com.lankydan.service.MyServiceImpl.doStuff - doStuff needed to wa
28-08-2017 13:32:18.549 [main] ERROR com.lankydan.service.MyServiceImpl.doStuff - doStuff encountered
```

请注意，即使 TRACE 和 DEBUG 级别的消息被发送到 logger ，但它们并没有被显示是因为它们的级别低于 INFO 级别。

Tocy

翻译于 2017/09/11 10:56

顶 0

```
logging.level.root=info
logging.pattern.console=%d{dd-MM-yyyy HH:mm:ss.SSS} %magenta([%thread]) %highlight(%-5level) %logger.%
```

当以这种形式完成时，logback.xml 文件是不要求的，同时可以看出，这种配置对于简单设定更加简短实用。

当你使用 Spring Boot ，添加自己的 logback.xml 时，LogBack 的默认配置将会被覆盖。假如你希望包含 Spring Boot 的配置，你可以在标签内添加如下内容。

```
<include resource="org/springframework/boot/logging/logback/base.xml"/>
```

参考 Spring Boot docs – Configure Logback for logging 获取关于这方面的更多信息。

假如你希望为某些类以不同于根级别的消息类型来记录日志消息，你可以为此类定义自己的日志。这样将允许你为特别的类设定日志级别，就像为这个类定制了一些其他属性。下面是你如何为单独的一个类定义日志。

```
<logger name="com.lankydan.service.MyServiceImpl" level="debug">
    <appender-ref ref="STDOUT" />
</logger>
```

vampire88

翻译于 2017/09/12 15:46

假如你继续运行代码片段，并且根日志已经定义，它将产生输出：

```
27-08-2017 17:02:10.248 [main] DEBUG com.lankydan.service.MyServiceImpl.doStuff - doStuff needed to de
27-08-2017 17:02:10.248 [main] DEBUG com.lankydan.service.MyServiceImpl.doStuff - doStuff needed to de
27-08-2017 17:02:10.248 [main] INFO  com.lankydan.service.MyServiceImpl.doStuff - doStuff took input -
27-08-2017 17:02:10.248 [main] INFO  com.lankydan.service.MyServiceImpl.doStuff - doStuff took input -
27-08-2017 17:02:10.248 [main] WARN  com.lankydan.service.MyServiceImpl.doStuff - doStuff needed to wa
27-08-2017 17:02:10.248 [main] WARN  com.lankydan.service.MyServiceImpl.doStuff - doStuff needed to wa
27-08-2017 17:02:10.248 [main] ERROR com.lankydan.service.MyServiceImpl.doStuff - doStuff enc      ed
27-08-2017 17:02:10.248 [main] ERROR com.lankydan.service.MyServiceImpl.doStuff - doStuff enc      ed
```

设定 additivity="false"，由于根日志附加器和类级别附加器都写入日志文件，就会导致消息被输出两次。即使根级别是 ERROR，类级别设定为 DEBUG，对于 MyServiceImpl 类，它将全局覆盖它，导致根日志附加器为 DEBUG 级别。下面是包含此属性的代码，看起来应该像这样：

```
<logger name="com.lankydan.service.MyServiceImpl" additivity="false" level="debug">
    <appender-ref ref="STDOUT" />
</logger>
```

vampire88

翻译于 2017/09/12 16:22

另外一种可能的解决方案是只对类设置日志级别，并不写入日志（由于没用定义附加器）。这和上面的版本是等同的，但它使用了另外一个日志附加器（这里是根附加器）为它写入日志使之运行：

```
<logger name="com.lankydan.service.MyServiceImpl" level="debug"/>
```

假如两者中的任何一个解决方案被使用，返回的输出都是所期望的。

```
27-08-2017 16:30:47.818 [main] DEBUG com.lankydan.service.MyServiceImpl.doStuff - doStuff needed to de
27-08-2017 16:30:47.834 [main] INFO  com.lankydan.service.MyServiceImpl.doStuff - doStuff took input -
27-08-2017 16:30:47.834 [main] WARN  com.lankydan.service.MyServiceImpl.doStuff - doStuff needed to wa
27-08-2017 16:30:47.834 [main] ERROR com.lankydan.service.MyServiceImpl.doStuff - doStuff encountered
```

类级别的日志可以在 application.properties 文件中添加以下内容。

```
logging.level.com.lankydan.service.MyServiceImpl=debug
```

包级别的日志也可以简单地使用包名替代类名在日志标签中进行定义。

```
<logger name="com.lankydan.service" additivity="false" level="debug">
    <appender-ref ref="STDOUT" />
</logger>
```

```
<logger name="org.springframework.boot" level="debug">
    <appender-ref ref="STDOUT" />
</logger>
```

比较:

```
<logger name="org.springframework.boot.SpringApplication" level="debug">
    <appender-ref ref="STDOUT" />
</logger>
```

它打印出一个完整的不同的日志行数字。可能是数百而不是一行或两行，SpringApplication 的日志包含在 org.springframework.boot 日志中。

application.properties 中包级别的日志遵循同一种格式只是用包名替换类名。

```
logging.level.com.lankydan.service=debug
```

vampire88

翻译于 2017/09/12 16:50

顶 0

当您需要标记要记录的日志的输出文件夹时，可以通过配置文件重新定义属性，这是很方便的。

```
<property name="LOG_PATH" value="logs"/>
```

这是属性名为 LOG_PATH 的作用示例，并将使用目录 DEV_HOME / logs ，其中 DEV_HOME 是项目的根目录（至少我的是）。这可能不是将日志保存到现实中的最佳位置，但是对于本教程的需求，它是适合的。 LOG_PATH 是对默认的 Spring boot 日志记录设置非常重要的属性，还可以创建任何名称的属性。在整个配置的其余部分访问 LOG_PATH 的值，可以通过添加 $ {LOG_PATH} 来访问。

此配置还可以通过 application.properties 实现，因为 LOG_PATH 在 Spring Boot 中非常重要性。

```
logging.path=logs
```

```
ertyA=value
ertyB=${propertyA} # extra configuration if required
```

pertyA} 将被 propertyA 的值替换，从而允许 propertyB 使用它。

亚林瓜子

翻译于 2017/09/13 08:50

顶 0

使用 FileAppender 能够将日志保存到文件中。这是一个简单的日志追加程序，并将所有的日志保存到一个单一的文件，但是这样做可能会让文件变得非常大，所以你更有需要使用 RollingFileAppender 来进行切割，稍后我们再来看看。

```
<appender name="SAVE-TO-FILE" class="ch.qos.logback.core.FileAppender">
    <file>${LOG_PATH}/log.log</file>
    <encoder class="ch.qos.logback.classic.encoder.PatternLayoutEncoder">
        <Pattern>
            %d{dd-MM-yyyy HH:mm:ss.SSS} [%thread] %-5level %logger{36}.%M - %msg%n
        </Pattern>
    </encoder>
</appender>
```

这里没有太多配置。它与 ConsoleAppender 具有相同的结构，并添加了将日志消息保存到的文件。值得注意的是，我删除了在保存到文件时添加到编码器日志高亮配置，因为它将包含不想显示的字符，并且会使日志文件变得混乱。然后可以使用与前面显示的 STDOUT appender 相同的方法来引用这个 appender 。正如下面的代码引用片段：

```
<logger name="com.lankydan.service.MyServiceImpl" additivity="false" level="debug">
    <appender-ref ref="SAVE-TO-FILE" />
</logger>
```

亚林瓜子

翻译于 2017/09/13 08:57

顶 0