logback代理日志自动清理配置修改说明

appender

负责写日志的组件, 它有两个必要属性name和class。

```
<appender name="stdout" class="ch.qos.logback.core.ConsoleAppender"></appender>
<appender name="appLogAppender"
class="ch.qos.logback.core.rolling.RollingFileAppender"></appender>
```

name: appender名称

class: 指定appender的全限定名, 其默认有以下几种

1. **ConsoleAppender**: 日志输出到控制台, 全限定名: `ch.qos.logback.core.ConsoleAppender`
2. **FileAppender**: 日志输入到文件, 全限定名: `ch.qos.logback.core.FileAppender`
3. **RollingFileAppender**: 滚动记录文件, `FileAppender`的子类, 当符合条件（大小、时间）, 日志进行切分处理。全限定名: `ch.qos.logback.core.rolling.RollingFileAppender`

我们记录日志文件, 使用的就是RollingFileAppender

RollingFileAppender

```
<appender name="appLogAppender"
class="ch.qos.logback.core.rolling.RollingFileAppender">
    <file>${LOG_FOLDER}/${LOG_FILENAME}.log</file>
    <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
        <fileNamePattern>${LOG_FOLDER}/${LOG_FILENAME}-%d{yyyy-MM-dd}-
%i.log</fileNamePattern>
        <MaxHistory>60</MaxHistory>
        <timeBasedFileNamingAndTriggeringPolicy
class="ch.qos.logback.core.rolling.SizeAndTimeBasedFNATP">
            <maxFileSize>100MB</maxFileSize>
        </timeBasedFileNamingAndTriggeringPolicy>
    </rollingPolicy>
    <layout class="ch.qos.logback.classic.PatternLayout">
        <pattern>%d{yyyy-MM-dd HH:mm:ss} [ %thread ] - [ %-5level ] [
%logger{50}:%line ] - %msg%n</pattern>
    </layout>
</appender>
```

`RollingFileAppender`是`FileAppender`的子类，扩展了`FileAppender`，具有翻转日志文件的功能。

例如`RollingFileAppender`可以记录到名为`log.txt`文件的文件，并且一旦满足某个条件，就将其日志记录目标更改为另一个文件。

有以下子节点:

1. `<file>`：被写入的文件名，可以是相对目录，也可以是绝对目录，如果上级目录不存在会自动创建，没有默认值。
2. `<append>`：如果是 true，日志被追加到文件结尾，如果是 false，清空现存文件，默认是true。
3. `<rollingPolicy>`：当发生滚动时，决定`RollingFileAppender`的行为，涉及文件移动和重命名。
4. `<triggeringPolicy>`：告知`RollingFileAppender`何时激活滚动。

注意：

一般情况下，`<rollingPolicy>`和`<triggeringPolicy>`都可以用来配置滚动策略，两个节点**使用其中一种就行。**有一种特殊情况，当使用`<rollingPolicy>`节点并配置`FixedWindowRollingPolicy`滚动策略，就需要`<triggeringPolicy>`来配合使用，`<triggeringPolicy>`节点可以配置成`SizeBasedTriggeringPolicy`滚动策略。

rollingPolicy

```
<!-- 1. 基于时间的滚动策略 -->
<rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy" />
<!-- 2. 基于文件大小和时间滚策略 -->
<rollingPolicy class="ch.qos.logback.core.rolling.SizeAndTimeBasedRollingPolicy"
/>
<!-- 3. 基于文件个数的滚动策略 -->
<rollingPolicy class="ch.qos.logback.core.rolling.FixedWindowRollingPolicy" />
```

TimeBasedRollingPolicy

```
<rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
    <fileNamePattern>${LOG_FOLDER}/${LOG_FILENAME}-%d{yyyy-MM-dd}-
%i.log</fileNamePattern>
    <MaxHistory>60</MaxHistory>
    <timeBasedFileNamingAndTriggeringPolicy
class="ch.qos.logback.core.rolling.SizeAndTimeBasedFNATP">
        <maxFileSize>100MB</maxFileSize>
    </timeBasedFileNamingAndTriggeringPolicy>
</rollingPolicy>
```

基本属性：

`<fileNamePattern>`：必要节点，包含文件名及“%d”转换符。“%d”可以包含一个`java.text.SimpleDateFormat`指定的时间格式，如：%d{yyyy-MM} 表示按月滚动。如果直接使用%d，默认格式是 yyyy-MM-dd，表示按天滚动。 %i：当文件大小超过`maxFileSize`时，按照i进行文件滚动。

`<maxHistory>`：可选属性，控制保留的归档文件的最大数量，超出数量就删除旧文件。假设设置每天滚动，且maxHistory是365，则只保存最近365天的文件，删除之前的旧文件。注意，删除旧文件是，那些为了归档而创建的目录也会被删除。

`<totalSizeCap>`：可选属性，用来控制所有归档文件总的大小。当达到这个大小后，旧的归档文件将会被异步的删除。使用这个属性时还需要设置 `maxHistory` 属性。而且，`maxHistory` 将会被作为第一条件，该属性作为第二条件。

`<cleanHistoryOnStart>`：boolean类型，如果设置为 true，那么在 appender 启动的时候，归档文件将会被删除。默认的值为 false。归档文件的删除通常在轮转期间执行。但是，有些应用的存活时间可能等不到轮转触发。对于这种短期应用，可以通过设置该属性为 true，在 appender 启动的时候执行删除操作。

配置修改

1. 找到代理安装目录后, 进入配置文件目录 (`**/bftagent/config`)
2. 使用 `vi` 或者 `vim` 编辑配置 `logback.xml`
3. 找到 `appLogAppender` 内的 `<rollingPolicy>` 标签, 再找到 `<rollingPolicy>` 标签内的 `MaxHistory` 修改数值(当前数值**单位是天**), 这里修改的是 `bftagent-yyyy-MM-dd-i.log` 的保存天数
4. 找到 `hotUpdateLogAppender` 内的 `<rollingPolicy>` 标签, 再找到 `<rollingPolicy>` 标签内的 `MaxHistory` 修改数值(当前数值**单位是天**), 这里修改的是 `patch_update-yyyy-MM-dd-i.log` 的保存天数