1. Cấu hình Log4J bằng log4j2.properties

Khi không phát hiện file cấu hình nào trong đường dẫn, Log4J sẽ tự động ghi log theo thiết lập mặc định

\*Chèn ảnh log console khi không có file properties\*

----------

Để tự cấu hình cho Log4J theo ý của bản thân, ta cần thêm vào đường dẫn của chương trình 1 file cấu hình để chỉ cho Log4J biết nên ghi log như thế nào.

Có 4 định dạng chính của file cấu hình: .properties, .yaml, .json và .xml

Đối với những chương trình đơn giản, hoặc với những người mới làm quen với Java, sử dụng log4j.properties là phương pháp phù hợp và hiệu quả nhất

----------

Trước tiên, tạo file log4j2.properties trong đường dẫn src/main/resources  
Graphical user interface, text, application

Description automatically generated

----------

Log4J sử dụng Logger và Appender làm 2 thành phần chính để ghi log. Trong đó:

* Logger xử lí việc bắt các sự kiện và chuyển cho Appender tương ứng
* Appender nhận các sự kiện do Logger chuyển đến và ghi lại ở một điểm đến (console, file,...)

Trong file log4j.properties, ta sẽ cấu hình các Logger và Appender để có thể cho ra ghi các sự kiện theo ý muốn bản thân

----------

Ví dụ sau là một cấu hình giúp ghi các sự kiện ra màn hình console.

Timeline

Description automatically generated with low confidence

*Một file log4j2.properties đơn giản*

----------

Trong đó:

* appender.console.type : loại của Appender, trong ví dụ này là Console
* appender.console.name : tên của Console Appender
* appender.console.layout : thiết kế cách Console Appender sẽ ghi log
* appender.console.filter : cách lọc event của Console Appender
* rootLogger.level : level tối thiểu của của rootLogger
* rootLogger.appenderRef : chọn Appender sẽ được sử dụng cho Logger (rootLogger)

Lưu ý: Bất kì cấu hình nào cũng cần phải có rootLogger. Nếu chỉ có 1 Logger thì đó phải là rootLogger.

----------

Graphical user interface, text

Description automatically generated

*Kết quả màn hình console của một chương trình đơn giản*

----------

1. Log sự kiện ra file

Thay vì ghi log ra console, chúng ta có thể ghi log ra file riêng bằng cách cấu hình file log4j2.properties

Với file log4j2.properties ở phần trước, ta thêm phần cấu hình cho RollingFileAppender như sau:

A picture containing timeline

Description automatically generated

*Cấu hình cho file log4j2.properties để tạo file log*

----------

Trong đó:

* appender.rolling.type : loại của Appender, trong ví dụ này là RollingFile
* appender.rolling.name : tên của RollingFile Appender
* appender.rolling.fileName : tên của file mà Appender sẽ ghi ra
* appender.rolling.filePattern : định dạng tên file khi chia file log
* appender.rolling.layout : thiết kế cách RollingFile Appender sẽ ghi log
* appender.rolling.policies : chính sách chia file log
* appender.rolling.strategy: chiến lược chia file log
* logger.rolling.name : tên package mà logger sẽ hoạt động
* logger.rolling.level : level tối thiểu của logger
* logger.rolling.additivity : tính lặp lại của logger, dùng để tránh việc logger và rootLogger lặp lại cùng 1 nội dung
* logger.rolling.appenderRef : chọn Appender sẽ được sử dụng cho Logger

----------

Table

Description automatically generated with medium confidence

*File log và màn hình console khi chạy chương trình*

----------

1. Chia file log theo thời gian và dung lượng

Như đã giới thiệu, trong phần cấu hình RollingFile Appender, chúng ta có thể dùng appender.rolling.policies và appender.rolling.strategy để chia file log theo tùy ý.

Nguyên lý chia file:

* File log được xác định trong phần appender.rolling.fileName sẽ liên tục ghi cho đến khi bị buộc tách file (được qui định trong phần appender.rolling.policies)
* Phần log trong file trên sẽ được lưu vào một file khác theo định dạng qui định trong phần appender.rolling.filePattern
* Sau khi đã lưu trữ xong, file log được xác định trong phần appender.rolling.fileName sẽ bắt đầu ghi lại từ đầu, và tương tự lặp lại cho các lần ngắt tiếp theo

----------

Ví dụ ở phần trước đã giới thiệu 2 kiểu chia file cơ bản, đó là theo thời gian và theo dung lượng.

Timeline

Description automatically generated with medium confidence

*Một cấu hình chia file cơ bản*

----------

Trong đó:

* appender.rolling.policies.type : chọn chính sách dùng để chia file
* appender.rolling.policies.time.type : chọn chính sách để chia file theo thời gian
* appender.rolling.policies.time.interval : thời lượng cho 1 file, đơn vị là thành phần nhỏ nhất được qui định trong thành phần thời gian của appender.rolling.filePattern
* appender.rolling.policies.time.modulate : xác định có chia file ngay vào thời điểm ranh giới giữa các khoảng thời lượng với nhau hay không
* appender.rolling.policies.size.type : chọn chính sách để chia file theo dung lượng
* appender.rolling.policies.size.size : dung lượng tối đa của 1 file log
* appender.rolling.strategy.type : chọn chiến lược chia file
* appender.rolling.strategy.max : số file log tối đa được lưu trữ

----------

Để kiểm chứng, ta chạy chương trình ví dụ sau:

Text

Description automatically generated

----------

Ta sẽ được các file sau:

Graphical user interface, text, application

Description automatically generated

Chí tiết các file như sau:

Graphical user interface, text, application, Word

Description automatically generated

Graphical user interface, application, Word

Description automatically generated

----------

1. Qui định các log level

Các log level có một giá trị nguyên nhất định để qua đó có thể so sánh các log level với nhau. Các log level mặc định có các giá trị sau:

A screenshot of a computer

Description automatically generated with low confidence

----------

Ngoài ra, ta có thể tự tạo log level của bản thân trực tiếp qua chương trình

Ở ví dụ dưới đây, ta tạo một level mới có tên là ALERT, với giá trị là 250.

Text

Description automatically generated with medium confidence

----------

Sau khi chạy chương trình với file log4j2.properties với cấu hình ghi file tương tự như phần trước, ta có file log như sau

Text

Description automatically generated

References:   
<https://logging.apache.org/log4j/2.x/manual/configuration.html>

<https://logging.apache.org/log4j/2.x/manual/appenders.html>

<https://logging.apache.org/log4j/2.x/manual/customloglevels.html>

<https://howtodoinjava.com/log4j2/log4j2-properties-example/>

<https://www.ibm.com/support/pages/setting-log4j-log-file-rollover-frequency>