TRƯỜNG ĐẠI HỌC GIA ĐỊNH

KHOA CÔNG NGHỆ THÔNG TIN

\*\*\*\*\*



**TÀI LIỆU BÁO CÁO GIỮA KỲ MÔN HỌC**

**LẬP TRÌNH HƯỚNG ĐỐI TƯỢNG NÂNG CAO**

**ĐỀ TÀI: JAVA COMMOM LOGGING**

GVHD: Ths. LÊ HUỲNH PHƯỚC

SVTH: NGUYỄN HỒ HẢI ANH 2008110020 – K14DCPM01

LÝ HOÀI LÂM 2008110053 – K14DCPM01

XÍCH VĂN HẬU 2008110008 – K14DCPM01

HOÀNG VIỆT 2008110321 –K14DCPM02

Tháng 8 – Năm 2022

**MỤC LỤC**

[1 Giới thiệu Java Commom Logging 2](#_Toc110291250)

[1.1. Khái niệm Commom Logging 2](#_Toc110291251)

[1.2. Mục đích của commom logging trong java 4](#_Toc110291252)

[1.3. Commom logging giải quyết được vấn đề gì? 4](#_Toc110291253)

[2 Phân tích những yếu tố cơ bản commom logging 5](#_Toc110291254)

[2.1. Ghi log như thế nào cho đúng 5](#_Toc110291255)

[a. Phải lưu log những gì? 5](#_Toc110291256)

[b. Lưu log ở đâu? 5](#_Toc110291257)

[c. Lưu log bằng cách gì? 6](#_Toc110291258)

[2.2. Thành phần cơ bản của Log 6](#_Toc110291259)

[3 Cài đặt và thử nghiệm 7](#_Toc110291260)

[3.1. Cài đặt 7](#_Toc110291261)

[3.2. Thử nghiệm 9](#_Toc110291262)

[4 Tổng kết 9](#_Toc110291263)

[TÀI LIỆU THAM KHẢO 9](#_Toc110291264)

# **Giới thiệu Java Commom Logging**

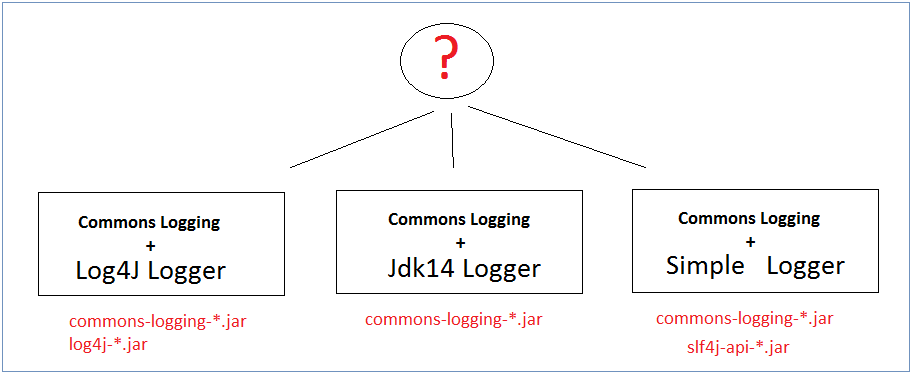
## **Khái niệm Commom Logging**

* Trong thực tế, Commom Logging (ghi nhật ký) được hiểu đại loại là một hành động lưu lại dấu vết, các thông tin được thông báo trong quá trình xử lý của một đoạn code, một chức năng hay một ứng dụng

Hiện nay mọi ứng dụng, hệ thống lớn nhỏ đều có thực hiện ghi log

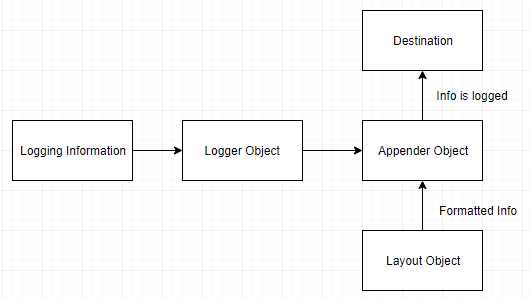
* Ví dụ: Như việc sử dụng điện thoại android, khi các bạn vào 1 ứng dụng bị lỗi, lập tức bị văng ra kèm theo 1 thông báo “send report” để người dùng có thể gửi log lỗi lên cho nhà phát triển để họ xem và sửa lỗi
* Còn trong Java, Commoms Logging là một thư viện cho phép bạn làm việc với một trong 3 bộ thư viện một cách dễ dàng hơn. Như vậy để viết một chương trình Log, một mình nó không thể là một Logger
* Commom Logging ra đời như một API(application programming interface-giao diện lập trình ứng dụng) chuẩn cho logging và với nền tảng hay framework khác nhau nó cũng có những biến thể nhất định như log4net và log4j

Như vậy chúng ta có khá nhiều bộ thư viện nguồn mở để thiết lập một hệ thống logging cho system của mình



Việc bạn chọn Logger nào không quan trọng chúng được xử lý một cách giống nhau với sự hỗ trợ của Commom Logging

* Ví dụ: Các thành phần chính của Log4j:
* Loggers: chịu trách nhiệm thu thập thông tin log
* Appenders: chịu trách nhiệm ghi log tới các vị trí cấu hình (file, cốnle..)
* Layouts: chịu trách nhiệm định dạng kết quả log

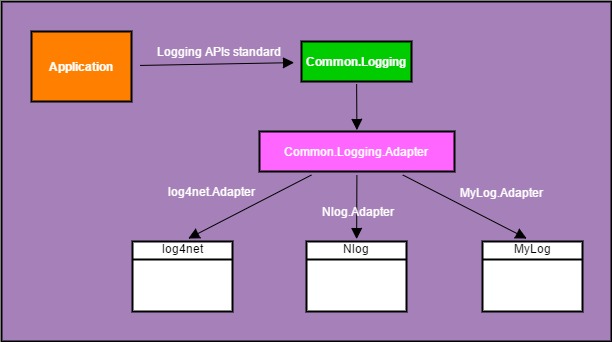


* Tính năng của Log4j : có độ bảo mật và tối ưu tốc độ, hỗ trợ nhiều output(file)
* Để sử dụng Log4j, chúng ta cần thực hiên theo các bước:
* Khai báo thư viện cần thiết cho Log4j
* Cấu hình Log4j
* Đặt câu lệnh log trong ứng dụng

## **Mục đích của commom logging trong java**

Khi đã là một developer trong quá trình xây dựng phần mềm, ắc hẳn chúng ta sẽ hiểu log có tầm quan trọng như thế nào khi những dòng code của chúng ta chạy(F5) nhưng không đem về kết quả mong muốn hay tệ hơn là gây hậu quả nghiêm trọng. Log cho ta biết những dòng code của chúng ta đang làm gì và có thể gặp vấn đề gì.Như vậy, để có thể debug và fix bug được hiệu quả chúng ta cần phải có log, một câu log đơn giản, nhưng giúp chúng ta nhìn ra vấn đề.

## **Commom logging giải quyết được vấn đề gì?**



Commom Logging sẽ cung cấp cho bạn một APIs chuẩn cho việc logging khi code, bạn sẽ không quan cần quan tâm đến việc sử dụng bộ thư viện logging nào hiệu quả để log. Từ đó việc logging của Dev trở nên dễ dàng hơn. Đến khi thật sự cần log ra file hay một dạng nào khác chúng ta mới cần quan tâm đến bộ thư viện log4net hay nlog tốt. Khi đó chúng ta sẽ dùng Commom Logging adapter để chuyển từ Commom Logging APIs qua log4net hay Nlog mà thôi. Mọi vấn đề đều đi đến một kết quả phải refactor code.

# **Phân tích những yếu tố cơ bản commom logging**

## **Ghi log như thế nào cho đúng**

### **Phải lưu log những gì**

Ghi log nhiều sẽ làm chúng ta khó kiểm soát, khi cần mà lại còn tốn dung lượng lưu trữ nữa.Còn lưu ít quá đôi khi cái lỗi mà chúng ta không ngờ tới rồi bỏ qua không lưu log đến khi xem lại không có. Vậy thì phải lưu đủ, không thừa, không thiếu và chúng ta nên lưu log ở một số ví trí sau

* Các tiến trình chạy ngầm

+ Lưu log lúc start và finish + thời gian xử lý xem có chuẩn với thời gian mà chúng ta đặt lịch không, tốc độ xử lý nhanh hay chậm

+ Lưu log lúc xử lý: Tiến trình đơn giản thì có thể lưu log trước và sau xử lý, còn tiến trình phức tạp, chúng ta nên ngắt các bước để lưu log để giám sát việc tiến trình đang xử lý đến bước nào và tiến trình xử lý nhiều có khả năng gây tải hệ thống, cần lưu log theo dõi thông tin server hiên tại RAM, CPU để tối ưu hóa hệ thống cũng như server phù hợp

* Khi giao tiếp với hệ thống khác (gọi vào và gọi ra). Cần lưu log các đoạn thực hiện gọi api/service: các thông tin api, thời gian xử lý, người request
* Các khối try catch excption: Catch thường dùng để bắt những lỗi mà chúng ta không biết nên sẽ có Log đi kèm thông tin của lỗi đó

### **Lưu log ở đâu?**

* Lưu log vào file trong hệ thống
* Lưu log vào cơ sở dữ liệu
* Cách làm này sẽ giúp tiết kiệm được tài nguyên hệ thống, log một cách dễ dàng hơn, rất thuận tiện trong việc giám sát log

### **Lưu log bằng cách gì?**

Bằng cách print ra màn hình những gì mong muốn nhưng cơ chế print là cơ chế thao tác tuần tự, dẫn đến chương trình, module tốn rất nhiều thời gian để xử lý dẫn đến chậm. Tùy theo mục đích và yêu cầu giám sát log để có thể chọn nơi lưu log, cách lưu log thích hợp cho dự án của bạn. Khi lưu log chúng ta cần phải tuân thủ 1 số nguyên tắc sau

Lưu log theo cấu trúc nhất định (lưu theo thời gian ngày tháng và phân theo loại log). Đảm bảo chương trình vẫn chạy bình thường khi quá trình ghi log xảy ra lỗi.

Thật may là hiện nay, hầu hết các framework hiện nay đều được cung cấp chức năng ghi log tự động, tuy nhiên bên cạnh việc đọc hiểu bug thì đọc log cũng là một kỹ năng rất quan trọng. Vd: log4j (phổ biến)

Với câu lệnh ghi log đơn giản. Tuy nhiên đằng sau nó là cả một hệ thống xử lý. Nội dung ghi log sẽ cho vào 1 hàng đợi, tiến trình trên sẽ quét hàng đợi để xử lý từng lệnh, giảm thiểu tối đa ảnh hưởng t/gian.

## **Thành phần cơ bản của Log**

* **Thời gian log:** Cho chúng ta biết được, khi nào xảy sự việc không mong muốn. Tại thời điểm này, tốt nhất chúng ta sẽ lưu thêm cả những thông tin đầu ra, đầu vào hàm để có thể hiểu rõ hơn về lý do vì sao xảy ra lỗi
* **Cấp độ log:** chúng ta thông thường có 6 cấp độ như sau

|  |  |
| --- | --- |
| **Cấp độ log (thứ tự giảm dần)** | **Ghi chú** |
| FATAL | Là những lỗi khá nghiêm trọng, có thể làm chết luôn hệ thống |
| ERROR | Là những lỗi có khả năng xảy ra |
| WARN | Là sự kiện đáng lẽ không nên xảy ra, nhưng đã xảy ra và chúng ta không biết có khả năng sự kiện này có lỗi gì không |
| DEBUG | Là những gì cần thiết để fix, nếu có bug sau này tại các vị trí đó, thông thường là các biến đầu vào, đầu ra của hàm |
| TRACE (giống debug) | Chúng ta cần log lại xem đã đi qua hàm nào, điều kiện để đi qua những hàm đó |
| INFO | Viết những thông tin cần thiết ra Log |

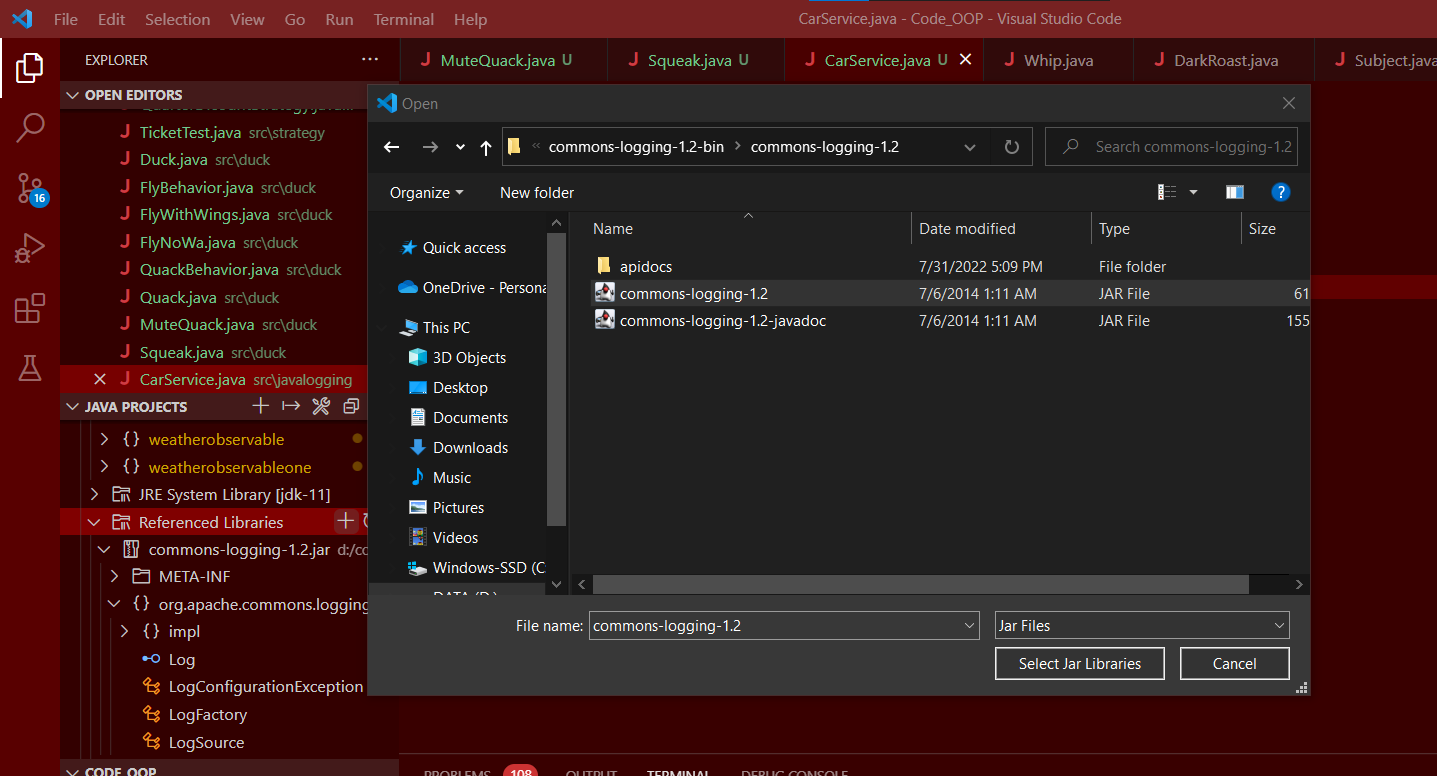
* **Vị trí Log:** Giúp chúng ta thực hiện debug và không biết lỗi đó xuất phát từ đâu. Giả sử chúng ta thực hiện một funcition và API không trả về bất cứ thứ gì, như vậy chúng ra chỉ cần xem lại log xem hàm nào là hàm có vấn đề và đó là mốc đầu tiên mà chúng ra sẽ kiểm tra
* **=> Những thông tin này là cơ bản cho mọi yêu cầu cần log, bất kể file log hay runtime log**

# **Cài đặt và thử nghiệm**

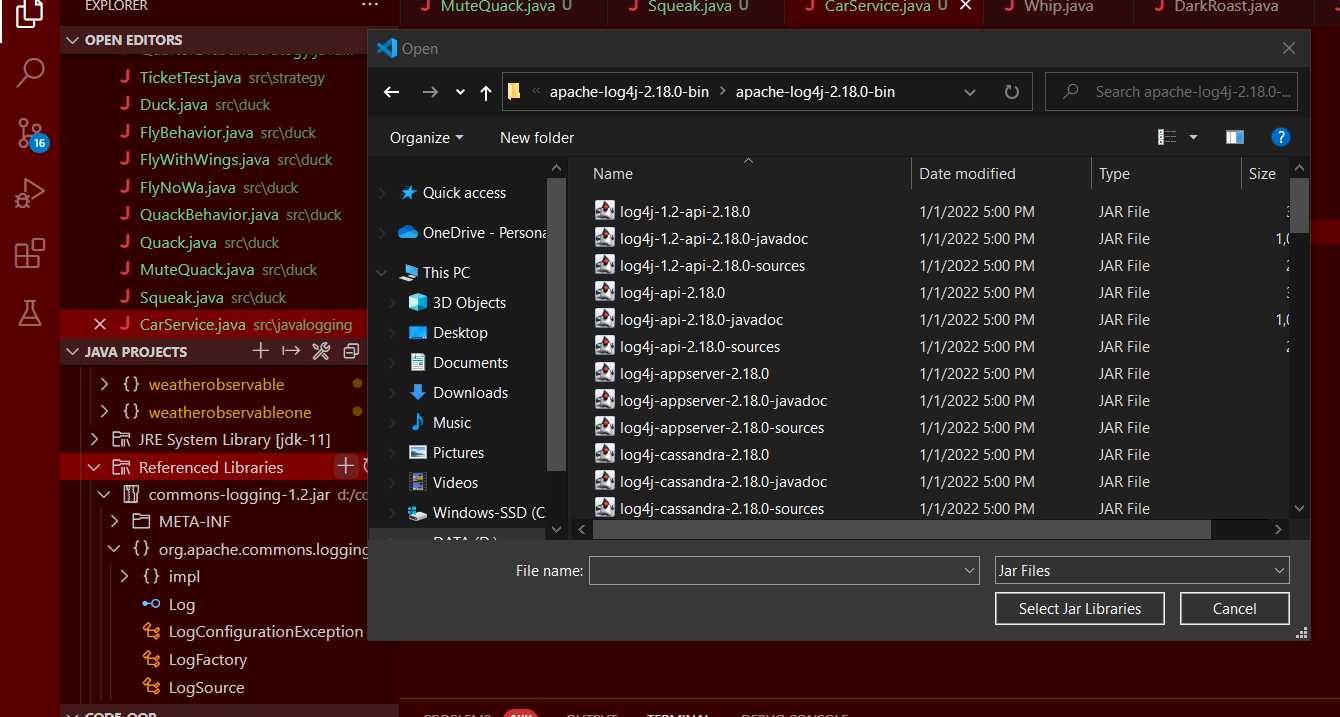
## **Cài đặt**

Bạn chỉ cần tải 1 file.jar về máy. Sau đó sao chép tệp này vào libs cuối cùng nó hoạt động

Thư viện chính:

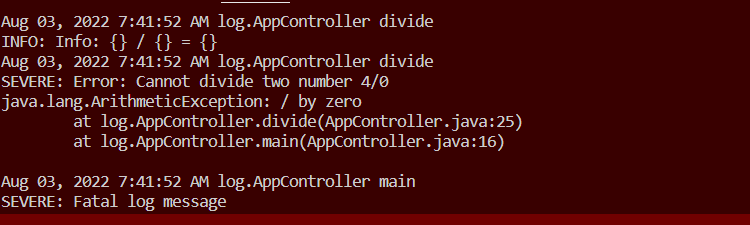


Thư viện hỗ trợ:



## **Thử nghiệm**





# **Tổng kết**

Nếu bạn đang phát triển một thư viện, tôi nghĩ rằng Commons Logging vẫn là lựa chọn tốt nhất để ghi nhật ký nội bộ. Nó cung cấp một giao diện nhất quán và thật hợp lý khi mong đợi rằng người tiêu dùng thư viện của bạn đã có JAR cầu nối mà họ cần. Bên cạnh đó sẽ có những ưu điểm

* giúp theo dõi dòng chảy của chương trình
* giúp nắm bắt bất kỳ lỗi nào có thể xảy ra
* cung cấp hỗ trợ chẩn đoán và gỡ lỗi vấn đề

# **TÀI LIỆU THAM KHẢO**

**Link tài liệu**

1. [**https://cafedev.vn/tu-hoc-java-ghi-loglogging-trong-java/**](https://cafedev.vn/tu-hoc-java-ghi-loglogging-trong-java/)
2. [**https://openplanning.net/10151/java-commons-logging#a10385**](https://openplanning.net/10151/java-commons-logging#a10385)
3. [**https://blog.kdgregory.com/2020/06/a-history-of-java-logging-frameworks-or.html**](https://blog.kdgregory.com/2020/06/a-history-of-java-logging-frameworks-or.html)
4. [**https://truonggiathien.com.vn/logging-la-gi-gioi-thieu-java-logging/#Cac\_cau\_hoi\_ve\_Logging\_La\_Gi\_8211\_Gioi\_Thieu\_Java\_Logging**](https://truonggiathien.com.vn/logging-la-gi-gioi-thieu-java-logging/#Cac_cau_hoi_ve_Logging_La_Gi_8211_Gioi_Thieu_Java_Logging)