# TỔNG QUAN LOGGING

Mặc dù [unit test](https://nmhblog.wordpress.com/2010/08/19/unit-test/) có thể giúp hạn chế lỗi khi phát triển phần mềm, với những chương trình phức tạp, sự cố là không thể tránh khỏi, nghĩa là vẫn phải có debug. Và việc này sẽ đơn giản hơn rất nhiều nếu trạng thái của chương trình tại các thời điểm khác nhau được ghi lại. Ngoài ra, các thông tin này có thể giúp cải tiến chương trình về giao diện sử dụng, tốc độ, v.v. Đó là lí do chúng ta có các logging framework.

# GIỚI THIỆU VỀ LOGGING

## Logging là gì?

Logging là 1 công cụ cần thiết mỗi lập trình viên. Nó giúp lập trình viên nhận dạng vấn đề nhanh hơn bằng cách show lên trạng thái của mỗi ứng dụng tại bất kỳ thời điểm nào , nó rất quan trọng sau khi triển khai

Logging thường tự động ghi các sự kiện và cấu trúc dữ liệu về sự thực thi của một chương trình phần mềm để cung cấp một lộ trình kiểm tra. Developers, testers and support personal thường sử dụng ghi các thông tin và dữ liệu để xác định phần mềm có vấn đề cho mục đích kiểm tra và gỡ lỗi.

Logging luôn luôn bao gồm gửi dữ liệu đến monitoring application hoặc write a messages text đến file.

Logging được chia ra làm 3 phần chính:  Logger, Formatter and the Handler (Appender).

## Logging framework

Logging framework là một computer data logging package cho Java flatform.

# NỘI DUNG

## THÀNH PHẦN LOGGING

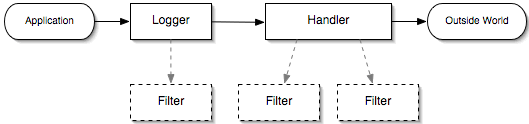


Figure1: Logging Component

Giải thích: Một ứng dụng yêu cầu logger log messages thì nhiệm vụ của logger là đưa các event tới appenders phù hợp appender sẽ chọn một simple handler để ghi dữ liệu.

* Có các loại handler:
* StreamHandler
* ConsoleHandler
* FileHandler
* SocketHandler
* MemoryHandler

Fitter là lọc các level. Sau khi lọc xong ghi dữ liệu ra bên ngoài.

* Logger: Chịu trách nhiệm lấy event và đưa chúng tới appenders phù hợp
* Appenders: Chịu trách nhiệm cho hành động ghi các log event đến đích. Appenders sử dụng Layout để để định dạng event trước khi gửi chúng đến một Output.
* Layouts: Cũng giống như formatters, chịu trách nhiệm cho việc chuyển đổi và định dạng dữ liệu trong một log event

Khi một logger log một event, nó sẽ chuyển nó đến appender phù hợp. Appenders sau đó sẽ định dạng log entry sử dụng layout trước khi nó gửi nó đến console, đến file hoặc một địa chỉ đích khác

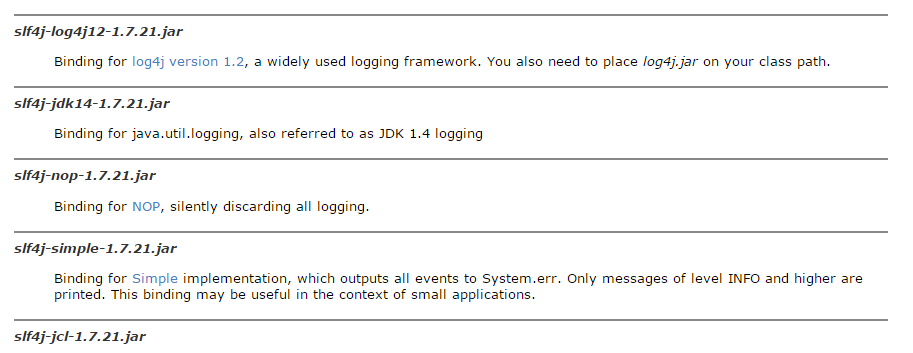
## Các loại logging framework.

Logging trong java yêu cầu sử dụng một hoặc nhiều logging framework. Một vài framework cung cấp object, method và cấu hình chắc chắn để truyền log messages. Java cung cấp một framework mặt định trong java.util.logging package. Có 3 framework bao gồm:

* Log4j
* Logback
* Tinylog.
* Có những packages khác như là SLF4J và Apache Commons Logging. Cung cấp một lớp trừu tượng cái mà tách mã code của bạn từ logging framework bên dưới cho phép bạn lựa chọn giữa các logging frameworks.

Abstaction layers giống như SLF4L tách ứng dụng của bạn từ logging framework . Một ứng dụng có thể chọn bắt một logging framework cụ thể ( như java.util.logging, Log4j, or Logback) tại thời điểm run time bằng cách đưa thêm những framework này lên application’s classpath. If như một logging framework không tồn tại trên classpath thì lớp trừu tượng sẽ hủy log call.

The Simple Logging Facade for Java (SLF4J): Được được dung như một mặt nạ hoặc một sự trừu tượng cho các logging framework khác nhau. Cho phép end-user có thể kết nối đến logging framework phù hợp tại thời điểm triển khai.



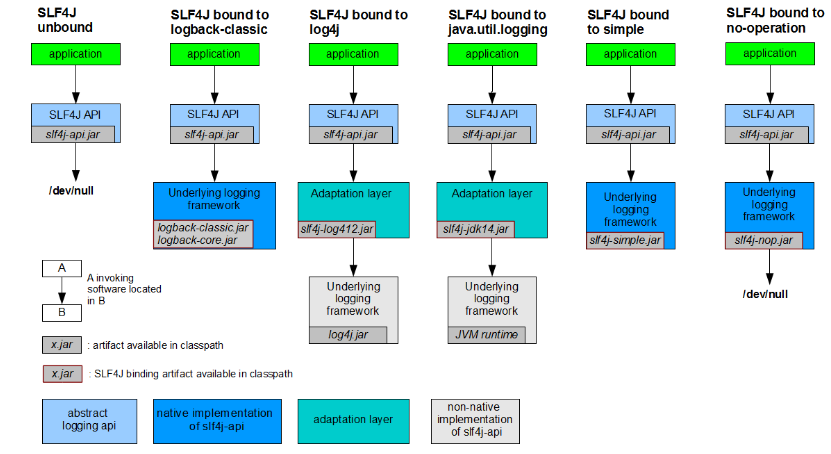
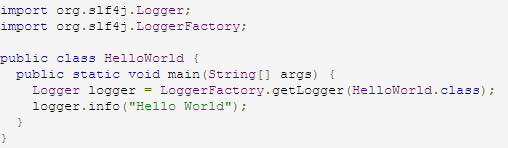


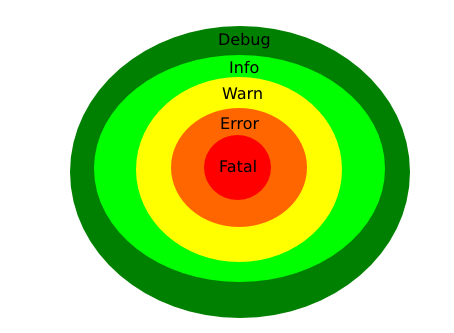
Figure2: SLF4J binding là một liên kết cứng tại thời điểm running time

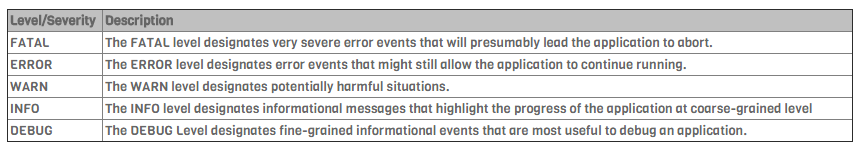
Sử dụng một và chỉ một logging framework cụ thể.



Để thực hiện đoạn code này ta phải download slf4j distribution về rồi giải nén ra sau đó giải nén lấy file slf4j-api-1.7.21.jar đến đường dẫn lớp.

Severity/Level: Tùy vào những framework khác nhau mà có những level nhiều hay ít nhưng sẽ có những level chung.





## So sánh giữa khác logging framework

# TÌM HIỂU VỀ LOG4J.

## Giới thiệu

Log4j là một thư viện cái mà được chỉ định trong logging.

Log4j đáng tin cậy, nhanh và logging framework linh động hiệu quả cái mà được phân phối dưới phần mềm apache license.  
Log4j đã được có trên ngôn ngữ c, c++, c#, rupy, python, ..

## Nội dung

### Thành phần :

* Log4j có 3 thành phần:
* Loggers: Chịu trách nhiệm lấy thông tin logging
* Appender: Chịu trách nhiệm đưa các thông tin logging đến các đích thích hợp
* Layout: Chịu trách nhiệm định dạng thông tin logging trong các kiểu khác nhau

### Tính năng

* Nó là một luồng an toàn
* Nó thì tối ưu hóa ứng dụng
* Nó thì dựa trên hệ thống cấp bậc logger name
* Nó hỗ trợ nhiều appenders ngõ ra trên mỗi logger.
* Nó hỗ trợ toàn cầu
* Một hành vi logging có thể cài đặt tại thời điểm runtime sử dụng a cấu hình file
* Đầu ra của logging có thể dễ dàng thay đổi bằng cách kế thừa từ lớp layout.

### Kiến trúc

Log4j theo một lớp kiến trúc nơi mà mỗi lớp cung cấp nhiều đối tượng khác nhau để thực hiện những tác vụ khác nhau. Lớp kiến trúc này tạo ra bản thiết kế linh động và dễ dàng mở rộng trong tương lai.

Có 2 kiểu đối tượng trong log4j framework.

* Core Object: Đây là đối tượng có tính bắt buộc trong framework, được yêu cầu sử dụng trong framework.
* Support Object: Đây là những đối tượng tùy chọn trong framework. Hỗ trợ core oject để thực hiện thêm một số tasks quan trọng
* Core object

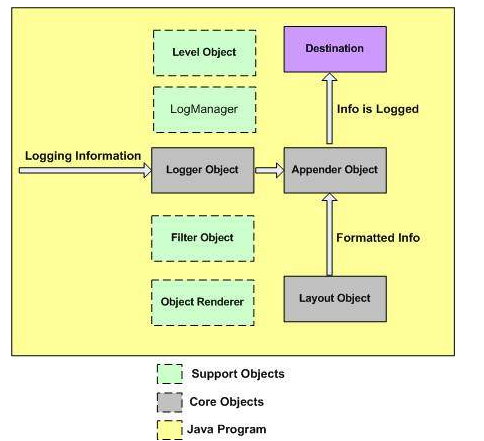


Figure 3: show các thành phần của log4j

Nhìn vào hình thì có 3 thành phần chính đó là logger, appender, layout

Còn filter, logManager, level là những thành phần hỗ trợ cho logger.

* Support object
* Level Object: Định nghĩa đối tượng ưu tiên và chi tiết của thông tin logging có 5 level của logging được định nghĩa trong API: OFF, DEBUG, INFO, ERROR, WARN, FATAL, và ALL.
* Filter Object: Được sử dụng để phân tích thông tin logging và tạo các quyết định tiếp theo nơi mà thông tin này có được log hay không. Một đối tượng appender có nhiều đối tượng filter liên kết với chúng. Nếu một thông tin logging được đưa đến appender thì tấc cả các đối tượng filter liên kết với appender này đề chấp thuận thông tin logging trước khi có thể đưa nó đến đích.
* Object Renderer: Đối tượng này cung cấp một chuỗi trình bày của những đối tượng khác nhau đã đưa đến logging framework. Đối tượng này sử dụng ở lớp Layout để chuẩn bị thông tin logging cuối cùng được xuất.
* LogManager: Đối tượng quản lý logging framework. Nó chịu trách nhiệm đọc các đối số cấu hình file khởi tạo từ một file cấu hình hệ thống lớn hoặc cấu hình lớp.

### Cấu hình

Chúng ta đã giới thiệu các thành phần chính của log4j. Sau đây sẽ giới thiệu làm như thế nào để cấu hình các thành phần chính sử dụng cấu hình file. Cấu hình file thì cấu hình cho Level, định nghĩa appender và đối tượng cụ thể Layout trong file cấu hình.

Log4j.properties là một cấu hình file cái mà lưu trữ cặp key-value. Mặc định LogManager sẽ tìm kiếm tên file log4j.properties trong classpath.

Level được định nghĩa là DEBUG, tên appender là X

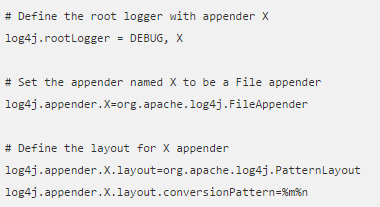


Figure4: File cấu hình tên

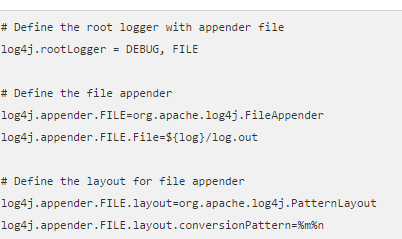


Figure 5: File cấu hình file

Level định nghĩa là DEBUG, DEBUG sẽ appender tên file đến nó

Appender FILE thì được định nghĩa như org.apache.log4j.FileAppender . Nó sẽ được viết đến file log.out vị trí trong thu mục log.

Định nghĩa Layout pattern là %m%n có nghĩa là xuất logging messages.

Mỗi đối tượng Appender đều có các thuộc tính khác nhau gắn kết với nó và các thuộc tính này cho biết hành vi của đối tượng đó.

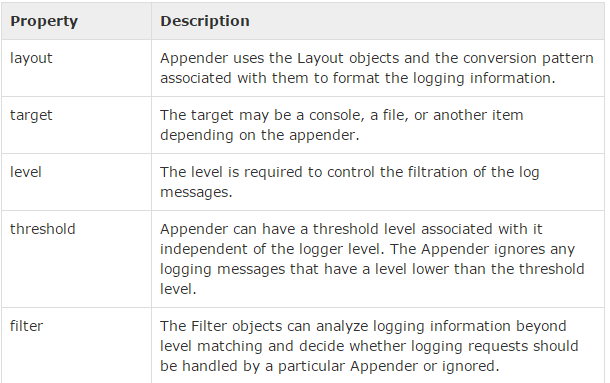


Table1: Các proberties of Appender

Chúng ta có thể thêm một đối tượng appender vào một logger bằng cách cài đặt cấu hình file như bên dưới.

log4j.logger.[logger-name]=level, appender1,appender..n

Chúng ta có thể thêm một appender vào chương trình bằng dòng lệnh sau:



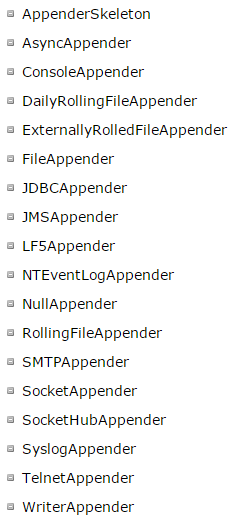


Table 2: Tấc cả các Appender

Chương trình ví dụ:

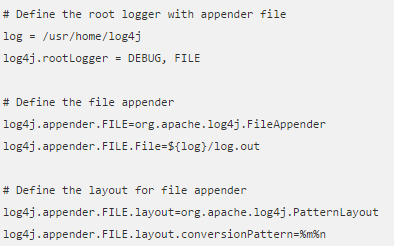


Figure 6: Ví dụ trên cấu hình file

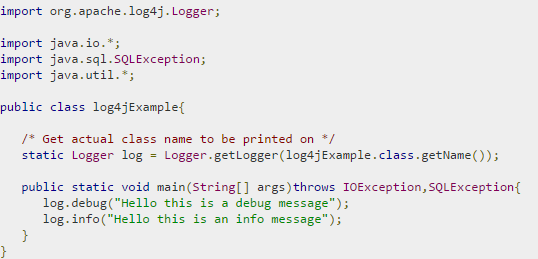


Figure 7: Ví dụ trên chương trình

### Phương thức logging

Lớp logger cũng cấp khá nhiều phương thức để xử lý hoạt động của logging. Lớp logger không cho phép chúng ta khỏi tạo một logger instance mới nhưng nó cung cấp hai phương thức tĩnh để lấy một đối tượng logger

* public static Logger getRootLogger();
* public static Logger getLogger(String name);

Phương thức public static Logger getRootLogger() trả về một logger instance và nó không có tên

Phương thức public static Logger getLogger(String name) trả về một logger instance và nó có tên



Lớp logger class có những phương thức bên dưới để xuất thông tin logging.



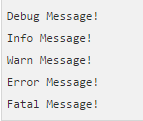
Table 3: Phương thức của lớp logger.

Ví dụ:



Figure 8: Ví dụ các phương thức của lớp logger

Khi chúng ta complie hoặc run thì kết quả như sau:



### Logging level

* Làm như thế nào để level làm việc.

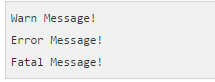
Một log yêu cầu level p trong một logger với level q thì được cho phép nếu p >=q.

Chúng ta có: ALL < DEBUG < INFO < WARN < ERROR < FATAL < OFF.

Ví dụ:



Khi chúng ta chạy thì kết quả sẽ là:



Cài đặt level sử dụng cấu hình file

Log4j cung cấp bạn cấu hình file dựa trên cài đặt level trong đó bạn sẽ hoàn toàn thay đổi từ mà nguồn khi mà bạn muốn thay đổi debug level

Ví dụ:

Chúng ta sử dụng phương thức log.setLevel(Level.WARN)

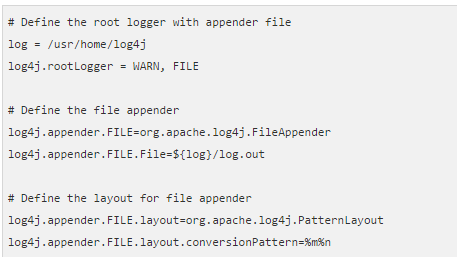


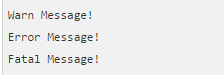
Figure 9: Cấu hình file

Sử dụng chương trình để cấu hình



Figure 10: Cấu hình chương trình

Kết quả chương trình:



## Log Formatting

Log4j cung cấp nhiều loại đối tượng Layout. Mỗi đối tượng trong số đó có thể định dạng dữ liệu logging theo những layout khác nhau. Nó có thể tạo ra đối tượng layout cái mà có thể định dạng dữ liệu logging trong một ứng dụng cụ thể

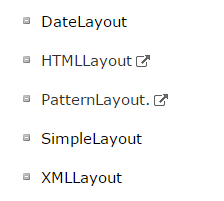
Tấc cả đối tượng layout nhận được một đối tượng LoggingEvent từ đối tượng Appender.

Đối tượng layout sau đó truy suất lấy các đối số messages từ LoggingEvent và áp dụng cho cho các ObjectReader phù hợp để có được những chuỗi trình bày của messages.

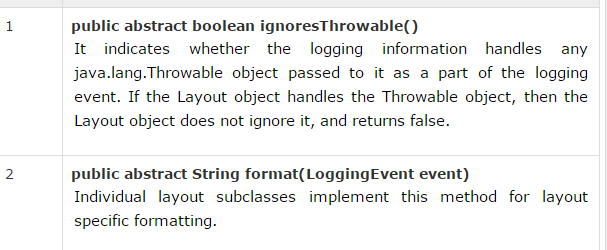
* Các kiểu Layout

Các lớp cao trong hệ thống phân cấp là lớp trừu tượng org.apache.log4j.Layout. Đây là lớp cơ sở cho tấc cả các lớp trong API log4j.

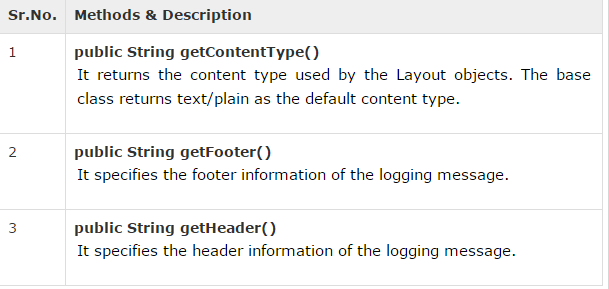
Lớp Layout thì được định nghĩa như một lớp trừu tượng trong một ứng dụng. Chúng ta không bao giờ sử dụng lớp này để điều khiển. Thay vì đó ta làm việc với các lớp con.



* Các phương thưc Layout



Ngoài những lớp trừu tượng thì layout cung cấp những lớp implement cho phương thức.



# THỰC HIỆN TRÊN NETBEANS VÀ ECLIPSE

## Thực hiện trên netbeans

### Thêm thư viện log4j

Bước 1: Đầu tiên chúng ta download thư viện từ <https://logging.apache.org/log4j/1.2/download.html>

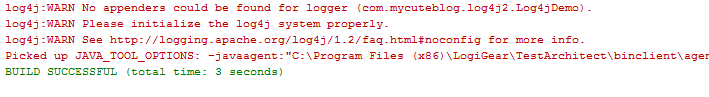
Bược 2: click chuột phải vào libraries -> add JAR/Folder-> đường dẫn tới file java vừa download về



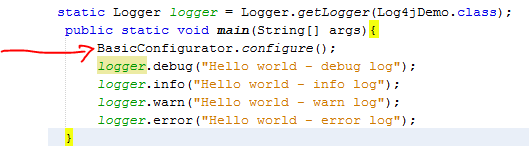
Sau khi thư viện lúc này ta có thể sử dụng log4j.

### Cấu hình cho log4j

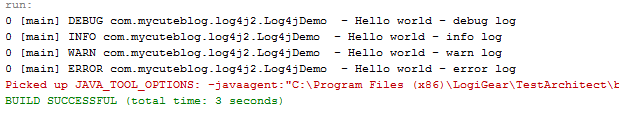
Nếu chúng ta không cấu hình cho log4j thì không thể thực hiện log ra file hay màn hình console được.



* Có thể chúng ta sử dụng cấu hình mặc định:

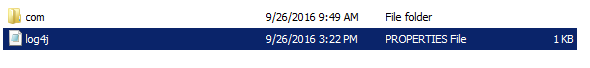


Kết quả xuất màn ra console là:



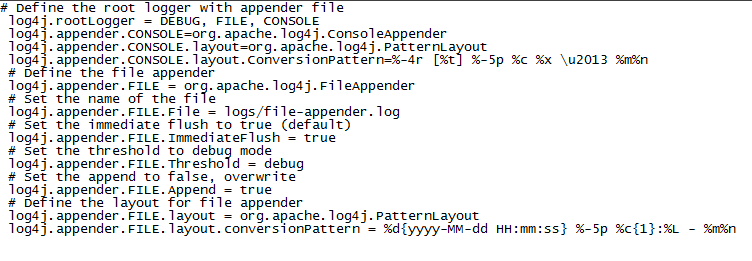
* Cấu hình bằng file properties

Ta tạo một file properties và cài đặt cấu hình. Sau khi cấu hình xong chúng ta copy file vào trong thư mục src của chương trình bạn

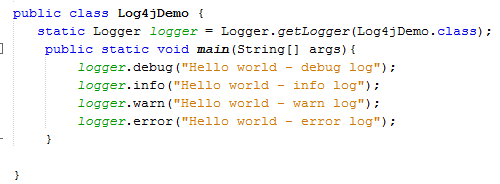


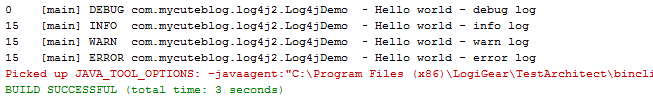
File log4j.properties nằm trong thu mục src

Đây là một file cấu hình.



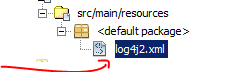
Sau đó ta thực hiện chạy chương trình



Kết quả: 

* Nếu sử dụng Maven

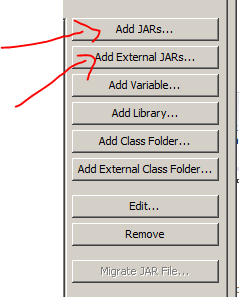
Ta sẽ thêm file cấu hình vào thu mục src/main/resources



## Thực hiện trên eclipse

### Thêm thư viện log4j

Click chuột phải vào  sau đó click chuột vào  sau đó ta muốn chọn thêm thư viện ở vị trí nào thì ta chỉ cần tìm thư mục chứa thư viện đó.



Sau khi thự hiện xong kết quả sẽ như sau:



### Cấu hình log4j

Tương tự như bên netbeans.

TÀI LIỆU THAM KHẢO VÀ ĐỀ MÔ

[1] <http://logging.apache.org/log4j/2.x/>

[2] <http://www.tutorialspoint.com/log4j/>