Logging trong Spring Boot

1. Tại sao không nên dùng `System.out.println()` để logging ?

Vì `System.out.println` có quá ít chức năng, và nó không thể hiển thị mức độ lỗi khác nhau, lưu log ra file, database...

2. Có rất nhiều thư viện Logging khác nhau như LogBack, Log4J, Log4J2...

- Log4J khá cũ rồi, khuyến cáo không nên dùng.
- LogBack được dùng mặc định bởi SpringBoot, chức năng ổn.
- Log4J2 cải tiến so với Log4J và có vài ưu điểm hơn LogBack như ghi log file Async

Để code dễ bảo trì, chúng ta nên dùng SLF4J là một Facade giao diện chung. Phía sau SLF4J sẽ dùng một thư viện Logging cụ thể. Khi dùng SLF4J chúng ta sẽ không phải viết lại code logging khi thay đổi thư viện Logging phía sau.

Bạn nên xem kỹ Youtube: Logback vs SLF4J vs Log4J2 - what is the difference? để hiểu khác biệt giữa các thư viện Logging

3. Sử dụng `SLF4J` và `Log4J2`

Trong file pom.xml, loại bỏ `spring-boot-starter-logging` ra khỏi `spring-boot-starter-web`

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-web</artifactId>
  <exclusions>
    <exclusion>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-logging</artifactId>
    </exclusion>
  </exclusions>
</dependency>
```

Thêm thư viện `spring-boot-starter-log4j2`

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-log4j2</artifactId>
</dependency>
```

và thêm thư viện Lombok

```
<dependency>
  <groupId>org.projectlombok</groupId>
  <artifactId>lombok</artifactId>
  <optional>true</optional>
</dependency>
```

4. Bổ xung annotation `@Slf4j` trước khai báo class

```
@Slf4j
public class APIController
```

Ý nghĩa của `@Slf4j` tương đương thêm thuộc tính `org.slf4j.Logger log` vào clas

```
public class APIController {
  private static final org.slf4j.Logger log = org.slf4j.LoggerFactory.getLogger(APICont
}
```

5. Cấu hình log4j2 bằng `log4j2.xml`

Trong thư mục resources tạo file log4j2.xml

```
resources
    ├── static
    ├── templates
    ├── application.properties
    ├── book.sql
    └── log4j2.xml <-- tạo file này
```

Có 3 khối chính trong log4j2.xml là:

1. Properties: cấu hình thuộc tính chung như thư mục chứa file log
2. Appenders: khai báo đầu ra của Log có thể là Console, File, Database...

3. Loggers: Các loại logger. Mỗi Logger có thể xuất ra nhiều Appenders.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Configuration monitorInterval="60">
  <Properties>
    <Property name="path">logs</Property>
  </Properties>
  <Appenders>
    <Console name="Console-Appender" target="SYSTEM_OUT">
      <PatternLayout>
        <pattern>
          %highlight{%-5p (%c{1}:%L) - %m%n%throwable}{FATAL=white, ERROR=red, WARN=yel
        </pattern>
      </PatternLayout>
    </Console>
    <File name="App-File-Appender" fileName="${path}/app.log">
      <PatternLayout>
        <pattern>
          [%-5level] %d{yyyy-MM-dd HH:mm:ss.SSS} [%t] %c{1} - %msg%n
        </pattern>
      </PatternLayout>
    </File>
    <File name="UnitTest-File-Appender" fileName="${path}/unittest.log">
      <PatternLayout>
        <pattern>
          [%-5level] %d{yyyy-MM-dd HH:mm:ss.SSS} [%t] %c{1} - %msg%n
        </pattern>
      </PatternLayout>
    </File>
    <File name="SpringBoot-File-Appender" fileName="${path}/springboot.log">
      <PatternLayout>
        <pattern>
          [%-5level] %d{yyyy-MM-dd HH:mm:ss.SSS} [%t] %c{1} - %msg%n
        </pattern>
      </PatternLayout>
    </File>
  </Appenders>
  <Loggers>
    <Logger name="org.springframework.web" level="debug" additivity="false">
      <AppenderRef ref="SpringBoot-File-Appender" />
      <AppenderRef ref="Console-Appender" />
    </Logger>
    <Logger name="vn.techmaster" level="debug" additivity="false">
      <AppenderRef ref="App-File-Appender" />
      <AppenderRef ref="Console-Appender" />
    </Logger>
    <Logger name="vn.techmaster.bookstore.FeignTest" level="debug" additivity="false">
      <AppenderRef ref="UnitTest-File-Appender" />
    </Logger>
    <Root level="info" additivity="false">
      <AppenderRef ref="Console-Appender" />
    </Root>
```

```
  </Loggers>
</Configuration>
```

5.1 Console-Appender

Có thể tuỳ chỉnh mầu sắc gán cho từng cấp độ log

```
<Console name="Console-Appender" target="SYSTEM_OUT">
  <PatternLayout>
    <pattern>
      %highlight{%-5p (%c{1}:%L) - %m%n%throwable}{FATAL=white, ERROR=red, WARN=yellow,
    </pattern>
  </PatternLayout>
</Console>
```

5.2 File Appender

Chúng ta có thể tạo ra nhiều file log cho những mục đích khác nhau, giúp dễ tìm ra lỗi hơn tuỳ theo phân vùng lỗi. Với mỗi một File Appender, lại có thể định dạng dữ liệu nhờ

```
<PatternLayout>
  <pattern>
    [%-5level] %d{yyyy-MM-dd HH:mm:ss.SSS} [%t] %c{1} - %msg%n
  </pattern>
</PatternLayout>
```

5.3 Phân luồng các dòng Log nhờ Loggers

Chuyên bắt các log phát sinh từ package `org.springframework.web` , bắt từ `level="debug"` , khi đã bắt được sự kiện rồi thì không ghi vào các dòng log khác nữa `additivity="false"`

```
<Logger name="org.springframework.web" level="debug" additivity="false">
  <AppenderRef ref="SpringBoot-File-Appender" />
  <AppenderRef ref="Console-Appender" />
</Logger>
```

Bắt các sự kiện log được tạo ra trong `class vn.techmaster.bookstore.FeignTest` , từ cấp độ `level="debug"`

```
<Logger name="vn.techmaster.bookstore.FeignTest" level="debug" additivity="false">
  <AppenderRef ref="UnitTest-File-Appender" />
</Logger>
```

Ví dụ nội dung file unittest.log

```
[INFO ] 2021-04-21 21:42:12.397 [main] FeignTest - Retry attempt 0
[INFO ] 2021-04-21 21:42:15.434 [main] FeignTest - Retry attempt 1
[INFO ] 2021-04-21 21:42:18.454 [main] FeignTest - Retry attempt 2
[INFO ] 2021-04-21 21:42:21.475 [main] FeignTest - Retry attempt 3
[ERROR] 2021-04-21 21:42:23.496 [main] FeignTest - Failed to execute after 4
[INFO ] 2021-04-21 22:13:42.571 [main] FeignTest - Retry attempt 0
[INFO ] 2021-04-21 22:13:45.602 [main] FeignTest - Retry attempt 1
[INFO ] 2021-04-21 22:13:46.711 [main] FeignTest - Call slow API success with 7 records returned
```

6. Các cấp độ log

Level	Value
OFF	0
FATAL	100
ERROR	200
WARN	300
INFO	400
DEBUG	500
TRACE	600
ALL	MaxValue