Logging

1.Công dụng

Công dụng chủ yếu của logging là ghi nhật ký trong quá trình chạy chương trình

2. Cách triển khai

2.1 Sử dụng Log4j2

```
import org.apache.logging.log4j.Logger;
import org.apache.logging.log4j.LogManager;
// [...]
Logger logger = LogManager.getLogger(LoggingController.class);
```

2.2 Sử dụng annotation Lombok

2.2.1 @Slf4j and @CommonsLog

SLF4J và Apache Commons Logging APIs cho phép chúng ta linh hoạt thay đổi khung ghi nhật ký mà không ảnh hưởng đến mã của chúng tôi.
Và chúng ta có thể sử dụng các chú thích @Slf4j và @CommonsLog của Lombok để thêm phiên bản trình ghi nhật ký phù hợp vào lớp của mình: org.slf4j.Logger cho SLF4J và org.apache.commons.logging.Log cho Ghi nhật ký Apache Commons.

```
@RestController
@Slf4j
public class LombokLoggingController {

    @RequestMapping("/lombok")
    public String index() {
        log.trace("A TRACE Message");
        log.debug("A DEBUG Message");
        log.info("An INFO Message");
        log.warn("A WARN Message");
        log.error("An ERROR Message");

        return "Howdy! Check out the Logs to see the output...";
    }
}
```

2.2.2 @Log4j2

Chúng ta có thể sử dụng chú thích @Log4j2 để sử dụng Log4j2 trực tiếp. Vì vậy, chúng ta thực hiện một thay đổi đơn giản cho LombokLoggingController để sử dụng @Log4j2 thay vì @Slf4j hoặc @CommonsLog:

```java
@RestController
@Log4j2
public class LombokLoggingController {

    @RequestMapping("/lombok")
    public String index() {
        log.trace("A TRACE Message");
        log.debug("A DEBUG Message");
        log.info("An INFO Message");
        log.warn("A WARN Message");
        log.error("An ERROR Message");

        return "Howdy! Check out the Logs to see the output...";
    }
}
```