WAS 구현

본 프로그래밍 과제는 Java 로 간단한 WAS(Web Application Server)를 만드는 것입니다.

과제를 수행할 때 Java API 문서, O'Reilly 의 <Java Network Programming>, 첨부한 예제 소스를 참고해도 좋습니다. 단 해당 코드를 이용할 경우, 읽기 좋지 않다고 생각하는 부분은 개선(리팩토링)해야 합니다.

(필독!) 과제 제출 시 유의 사항

- 소스를 압축 파일로 제출해주시고, 압축 파일 안의 README.md 파일에 아래 스펙의 구현 여부를 적어주세요.
- Git, SVN 을 사용한다 가정하고 꼭 필요한 파일만 제출하십시오.
- 제출한 소스는 압축을 푼 다음에 mvn clean package 했을 때, JUnit 을 수행하고, JAR 파일을 생성해야 합니다. 그리고 java -jar was.jar 해서 실행할 수 있어야 합니다.
- 아래 스펙을 모두 구현하지 못했더라도 최대한 작성하여 제출하면 됩니다. **전체 구현 완료 여부가 합격, 불합격의 절대 기준은 아닙니다.**
- **Java 표준 라이브러리 외 다른 네트워크 프레임워크(예, Netty)를 사용하지 말아주세요.**

스펙

1. HTTP/1.1 의 Host 헤더를 해석하세요.
    - 예를 들어, a.com 과 b.com 의 IP 가 같을지라도 설정에 따라 서버에서 다른 데이터를 제공할 수 있어야 합니다.
    - 아파치 웹 서버의 VirtualHost 기능을 참고하세요.
2. 다음 사항을 설정 파일로 관리하세요.
    - 파일 포맷은 JSON 으로 자유롭게 구성하세요.
    - 몇 번 포트에서 동작하는지
    - HTTP/1.1 의 Host 별로
        - HTTP_ROOT 디렉터리를 다르게
        - 403, 404, 500 오류일 때 출력할 HTML 파일 이름
3. 403, 404, 500 오류를 처리합니다.
    - 해당 오류 발생 시 적절한 HTML 을 반환합니다.
    - 설정 파일에 적은 파일 이름을 이용합니다.
4. 다음과 같은 보안 규칙을 둡니다.
    - 다음 규칙에 걸리면 응답 코드 403 을 반환합니다.

        - HTTP_ROOT 디렉터리의 상위 디렉터리에 접근할 때,
          예, http://localhost:8000/../../../../etc/passwd
        - 확장자가 .exe 인 파일을 요청받았을 때
    - 추후 규칙을 추가할 것을 고려해주세요.
5. logback 프레임워크 http://logback.qos.ch/를 이용하여 다음의 로깅 작업을 합니다.
    - 로그 파일을 하루 단위로 분리합니다.
    - 로그 내용에 따라 적절한 로그 레벨을 적용합니다.
    - 오류 발생 시, StackTrace 전체를 로그 파일에 남깁니다.
6. 간단한 WAS 를 구현합니다.
    - 다음과 같은 SimpleServlet 구현체가 동작해야 합니다.
        - 다음 코드에서 SimpleServlet, HttpRequet, HttpResponse 인터페이스나 객체는 여러분이 보다 구체적인 인터페이스나 구현체를 제공해야 합니다. **표준 Java Servlet 과는 무관**합니다.

```
public Hello implements SimpleServlet {
        public void service(HttpRequest req, HttpResponse res) {
                java.io.Writer writer = res.getWriter()
                writer.write("Hello, ");
                writer.write(req.getParameter("name"));
        }
}
```

    - URL 을 SimpleServlet 구현체로 매핑합니다. 규칙은 다음과 같습니다.
        - http://localhost:8000/Hello --> Hello.java 로 매핑
        - http://localhost:8000/service.Hello --> service 패키지의 Hello.java 로 매핑
    - 과제는 URL 을 바로 클래스 파일로 매핑하지만, 추후 설정 파일을 이용해서 매핑하는 것도 고려해서 개발하십시오.
        - 추후 확장을 고려하면 됩니다**. 설정 파일을 이용한 매핑을 구현할 필요는 없습니다.**
        - 설정 파일을 이용한 매핑에서 사용할 수 있는 설정의 예, {“/Greeting”: “Hello”, “/super.Greeting”: “service.Hello”}
7. 현재 시각을 출력하는 SimpleServlet 구현체를 작성하세요.
    - 앞서 구현한 WAS 를 이용합니다.
    - WAS 와 SimpleServlet 인터페이스를 포함한 SimpleServlet 구현 객체가 하나의 JAR 에 있어도 괜찮습니다.
        - 분리하면 더 좋습니다.
8. 앞에서 구현한 여러 스펙을 검증하는 테스트 케이스를 JUnit4 를 이용해서 작성하세요.