# 上机实践 10 程序文档

## 1、总览：

本节任务包括统一日志、分级别、动态配置、归档四个要求。

实现方法：

> 统一日志： logback 统一管理项目日志，同时客户端、服务器端分离

> 分级别：使用 logback 自带的日志级别管控功能，客户端、服务端初始设置为 info 级别

> 动态配置：定周期（30s）扫描配置文件读取最新的配置信息

> 归档：将不同级别日志输出至同一日志文件夹（clientlog\serverlog）下的不同子文件夹下

## 2、详细实现方法：

### 2.1 配置 logback

需要用到的组件包

slf4j-api-1.7.12.jar

logback-classic-0.9.29.jar

logback-core-0.9.29.jar

建立包的依赖即可；

### 2.2 编写 logback.xml 文件实现上述功能

#### （1）统一日志

本组通过比较众 log 开源库，选择 logback 进行本项目的日志统一管理

#### （2）简单的动态配置

```
<configuration debug="false" scan="true" scanPeriod="30 seconds">
```

scan 为 true 即可依据 scanPeriod 定时监测配置文件，可在其中进行更改信息

#### （3）分级别

Logback 中的根节点位 root

```
<root level="ERROR">
    <appender-ref ref="STDOUT" />
</root>
```

其余均为 logger 对象，有如下属性

```
 <logger name="clientlogback" level="INFO" additivity="true">
<appender-ref ref="clientErrorAppender"/>
<appender-ref ref="clientWarnAppender"/>
<appender-ref ref="clientInfoAppender"/>
  </logger>
```

通过名称可调用到 logger,例如:

mlogger = LoggerFactory.getLogger(“clientlogback”);

接着，即可调用此对象进行输出，例如

mlogger.info(“Test the info logback By .{}”,username);

//如果 username 是”FanLiang”,就会输出”Test the info logback By .FanLiang”.

此时，就是分级别的体现，level 会将其设置为某一级别，包括 ”trace\debug\info\warn\error” 5 个级别，强度递增。每一个 logger 只能输出比其级别高的日志信息，比如此时就不能输出：

mlogger.trace(“Test the info logback By .{}”,username);

OR

mlogger.debug(“Test the info logback By .{}”,username);

本项目中将 Client、Server 均默认设置为 Info 级别的日志输出

Appender-ref 为其具体的实现方法，一般包括*控制台*、*文件*、*按照一定规则输出*这三种形式，具体见下

### （4）归档

通过配置不同的 appender，可以实现不同的级别日志输出至不同的文件夹下，可以在进行归档，以下例子表示将 server 中的 INFO 级别的日志输出至对应的 ./serverlog/info/ 文件夹下

```
<appender name="serverInfoAppender"
class="ch.qos.logback.core.rolling.RollingFileAppender">
   <rollingPolicy
class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">

<fileNamePattern>${ServerLog_HOME}/info/%d{yyyy-MM-dd}.log</fileNamePat
tern>
```

```
      <MaxHistory>30</MaxHistory>
    </rollingPolicy>
    <encoder class="ch.qos.logback.classic.encoder.PatternLayoutEncoder">

<pattern>%d{HH:mm:ss.SSS} %thread %X{invokeNo} %logger{40} %msg%n</pattern
>
    </encoder>
    <filter class="ch.qos.logback.classic.filter.LevelFilter">
      <level>INFO</level>
      <onMatch>ACCEPT</onMatch>
      <onMismatch>DENY</onMismatch>
    </filter>
    <triggeringPolicy
class="ch.qos.logback.core.rolling.SizeBasedTriggeringPolicy">
          <MaxFileSize>10MB</MaxFileSize>
      </triggeringPolicy>
</appender>
```