|  |
| --- |
| App.java |
| **package** com.boot;  **import** org.slf4j.Logger;  **import** org.slf4j.LoggerFactory;  /\*\*  \* Hello world!  \*/  **public** **class** App {  **private** **final** **static** Logger ***logger*** = LoggerFactory.*getLogger*(App.**class**);  **public** **static** **void** main(String[] args) {  ***logger***.info("logback成功了");  ***logger***.debug("logback成功了");  ***logger***.error("logback成功了");  }  } |
| **logback.xml** |
| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <configuration debug=*"false"*>  <!--定义日志文件的存储地址 勿在 LogBack 的配置中使用相对路径 -->  <property name=*"LOG\_HOME"* value=*"/home"* />  <!-- 控制台输出 -->  <appender name=*"STDOUT"* class=*"ch.qos.logback.core.ConsoleAppender"*>  <encoder class=*"ch.qos.logback.classic.encoder.PatternLayoutEncoder"*>  <!--格式化输出：%d表示日期，%thread表示线程名，%-5level：级别从左显示5个字符宽度%msg：日志消息，%n是换行符 -->  <pattern>%d{yyyy-MM-dd HH:mm:ss.SSS} [%thread] %-5level %logger{50} - %msg%n</pattern>  </encoder>  </appender>  <!-- 按照每天生成日志文件 -->  <appender name=*"FILE"*  class=*"ch.qos.logback.core.rolling.RollingFileAppender"*>  <rollingPolicy class=*"ch.qos.logback.core.rolling.TimeBasedRollingPolicy"*>  <!--日志文件输出的文件名 -->  <FileNamePattern>${LOG\_HOME}/TestWeb.log.%d{yyyy-MM-dd}.log</FileNamePattern>  <!--日志文件保留天数 -->  <MaxHistory>30</MaxHistory>  </rollingPolicy>  <encoder class=*"ch.qos.logback.classic.encoder.PatternLayoutEncoder"*>  <!--格式化输出：%d表示日期，%thread表示线程名，%-5level：级别从左显示5个字符宽度%msg：日志消息，%n是换行符 -->  <pattern>%d{yyyy-MM-dd HH:mm:ss.SSS} [%thread] %-5level %logger{50} -  %msg%n</pattern>  </encoder>  <!--日志文件最大的大小 -->  <triggeringPolicy  class=*"ch.qos.logback.core.rolling.SizeBasedTriggeringPolicy"*>  <MaxFileSize>10MB</MaxFileSize>  </triggeringPolicy>  </appender>  <!-- 日志输出级别 -->  <root level=*"debug"*>  <appender-ref ref=*"STDOUT"* />  </root>  </configuration> |

2018-10-25 11:32:22.251 [main] INFO com.boot.App - logback成功了

2018-10-25 11:32:22.253 [main] DEBUG com.boot.App - logback成功了

2018-10-25 11:32:22.253 [main] ERROR com.boot.App - logback成功了

2:

logback.xml

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"UTF-8"*?>  <configuration>  <!-- %m输出的信息,%p日志级别,%t线程名,%d日期,%c类的全名,%i索引【从数字0开始递增】,,, -->  <!-- appender是configuration的子节点，是负责写日志的组件。 -->  <!-- ConsoleAppender：把日志输出到控制台 -->  <appender name=*"STDOUT"* class=*"ch.qos.logback.core.ConsoleAppender"*>  <encoder>  <pattern>%d %p (%file:%line\)- %m%n</pattern>  <!-- 控制台也要使用UTF-8，不要使用GBK，否则会中文乱码 -->  <charset>UTF-8</charset>  </encoder>  </appender>  <!-- RollingFileAppender：滚动记录文件，先将日志记录到指定文件，当符合某个条件时，将日志记录到其他文件 -->  <!-- 以下的大概意思是：1.先按日期存日志，日期变了，将前一天的日志文件名重命名为XXX%日期%索引，新的日志仍然是demo.log -->  <!-- 2.如果日期没有发生变化，但是当前日志的文件大小超过1KB时，对当前日志进行分割 重命名 -->  <appender name=*"demolog"*  class=*"ch.qos.logback.core.rolling.RollingFileAppender"*>  <File>log/demo.log</File>  <!-- rollingPolicy:当发生滚动时，决定 RollingFileAppender 的行为，涉及文件移动和重命名。 -->  <!-- TimeBasedRollingPolicy： 最常用的滚动策略，它根据时间来制定滚动策略，既负责滚动也负责出发滚动 -->  <rollingPolicy class=*"ch.qos.logback.core.rolling.TimeBasedRollingPolicy"*>  <!-- 活动文件的名字会根据fileNamePattern的值，每隔一段时间改变一次 -->  <!-- 文件名：log/demo.2017-12-05.0.log -->  <fileNamePattern>log/demo.%d.%i.log</fileNamePattern>  <!-- 每产生一个日志文件，该日志文件的保存期限为30天 -->  <maxHistory>30</maxHistory>  <timeBasedFileNamingAndTriggeringPolicy  class=*"ch.qos.logback.core.rolling.SizeAndTimeBasedFNATP"*>  <!-- maxFileSize:这是活动文件的大小，默认值是10MB，测试时可改成1KB看效果 -->  <maxFileSize>1KB</maxFileSize>  </timeBasedFileNamingAndTriggeringPolicy>  </rollingPolicy>  <encoder>  <!-- pattern节点，用来设置日志的输入格式 -->  <pattern>  %d %p (%file:%line\)- %m%n  </pattern>  <!-- 记录日志的编码:此处设置字符集 - -->  <charset>UTF-8</charset>  </encoder>  </appender>  <!-- 控制台输出日志级别 -->  <root level=*"info"*>  <appender-ref ref=*"STDOUT"* />  </root>  <!-- 指定项目中某个包，当有日志操作行为时的日志记录级别 -->  <!-- com.liyan为根包，也就是只要是发生在这个根包下面的所有日志操作行为的权限都是DEBUG -->  <!-- 级别依次为【从高到低】：FATAL > ERROR > WARN > INFO > DEBUG > TRACE -->  <logger name=*"com.lk"* level=*"DEBUG"*>  <appender-ref ref=*"demolog"* />  </logger>  </configuration> |