log4j基本使用方法

# Log4j的三个重要的组件构成

# 1、日志信息的优先级

日志信息的优先级从高到低有OFF、FATAL、ERROR、WARN、INFO、DEBUG、ALL。这里只讲常用的四个ERROR、WARN、 INFO、DEBUG，分别用来指定这条日志信息的重要程度。

# 2、日志信息的输出目的地

日志信息的输出目的地指定了日志将打印到控制台还是文件中。

# 3、日志信息的输出格式

而输出格式则控制了日志信息的显 示内容。

# 二、配置文件形式

①java特性文件（key=value）(log4j.properties)。

②直接在java代码中配置。

③XML文件格式的文件。

这里以第①种为主详细讲解，这也是我们用的最多的。

# 三、log4j.properties的详细配置

# 1、配置根Logger，其语法为：

log4j.rootLogger = [level],appenderName,appenderName, …

（1）level 是日志记录的优先级。通过在这里定义的级别，您可以控制到应用程序中相应级别的日志信息的开关。比如在这里定 义了INFO级别，则应用程序中所有DEBUG级别的日志信息将不被打印出来。

（2）appenderName就是指B日志信息输出到哪个地方。您可以同时指定多个输出目的地。

# 2、配置日志信息输出目的地Appender，其语法为：

log4j.appender.appenderName = fully.qualified.name.of.appender.class

log4j.appender.appenderName.option1 = value1

…

log4j.appender.appenderName.optionN = valueN

其中fully.qualified.name.of.appender.class提供的appender有以下几种值：

org.apache.log4j.ConsoleAppender控制台

org.apache.log4j.FileAppender 文件

org.apache.log4j.DailyRollingFileAppender每天产生一个文件

org.apache.log4j.WriterAppender将日志以流格式发送到任意指定的地方

org.apache.log4j.RollingFileAppender（文件大小到达指定尺寸的时候产生一个新的文件），

# 3、配置日志信息的格式（布局），其语法为：

log4j.appender.appenderName.layout = fully.qualified.name.of.layout.class

log4j.appender.appenderName.layout.option1 = value1

…

log4j.appender.appenderName.layout.optionN = valueN

其中fully.qualified.name.of.layout.clas提layout有以下几种值  
org.apache.log4j.HTMLLayout（以HTML表格形式布局）， org.apache.log4j.PatternLayout（可以灵活地指定布局模式）， org.apache.log4j.SimpleLayout（包含日志信息的级别和信息字符串）

org.apache.log4j.TTCCLayout（包含日志产生的时间、线程、类别等等信息）

4、Log4J采用类似C语言中的printf函数的打印格式格式化日志信息，打印参数如下：

①%m 输出代码中指定的消息

②%p 输出优先级，即DEBUG，INFO，WARN，ERROR，FATAL

③%r 输出自应用启动到输出该log信息耗费的毫秒数

④%c 输出所属的类目，通常就是所在类的全名

⑤%t 输出产生该日志事件的线程名

⑥%n 输出一个回车换行符，Windows平台为“rn”，Unix平台为“n”

⑦%d 输出日志时间点的日期或时间，默认格式为ISO8601.

也可以在其后指定格式，比如：%d{yyy-MMM-dd HH:mm:ss,SSS}，输出类似：2002-10-18 22:10:28,921

⑧%l 输出日志事件的发生位置，包括类目名、发生的线程，以及在代码中的行数。举例：Testlog4.main(TestLog4.java:10)

# 四、已完成的log4j.properties文件基础配置

### 设置###

log4j.rootLogger = debug,stdout,D,E

### 输出信息到控制抬 ###

log4j.appender.stdout = org.apache.log4j.ConsoleAppender

log4j.appender.stdout.Target = System.out

log4j.appender.stdout.Threshold = DEBUG

log4j.appender.stdout.layout = org.apache.log4j.PatternLayout

log4j.appender.stdout.layout.ConversionPattern = [%-5p] %d{yyyy-MM-dd HH:mm:ss,SSS}[%t:%r]%l%n%m%n

### 输出DEBUG 级别以上的日志到=E://logs/error.log ###

log4j.appender.D = org.apache.log4j.DailyRollingFileAppender

log4j.appender.D.File = E://logs/log\_

log4j.appender.D.DatePattern = yyyy-MM-dd'.log'

log4j.appender.D.Append = true

log4j.appender.D.Threshold = DEBUG

log4j.appender.D.layout = org.apache.log4j.PatternLayout

log4j.appender.D.layout.ConversionPattern = [%p][%-d{yyyy-MM-dd HH:mm:ss}][%t:%r] %l%n%m%n

### 输出ERROR 级别以上的日志到=E://logs/error.log ###

log4j.appender.E = org.apache.log4j.DailyRollingFileAppender

log4j.appender.E.File = E://logs/error\_

log4j.appender.E.DatePattern = yyyy-MM-dd'.log'

log4j.appender.E.Append = true

log4j.appender.E.Threshold = ERROR

log4j.appender.E.layout = org.apache.log4j.PatternLayout

log4j.appender.E.layout.ConversionPattern = [%p][%-d{yyyy-MM-dd HH:mm:ss}][%t:%r] %l%n%m%n