1、HTMLLayout

如果想生成一个HTML格式的文件，日志信息，那么可以使用 org.apache.log4j.HTMLLayout 格式化日志信息。HTMLLayout类扩展抽象org.apache.log4j.Layout类，并覆盖其基类的 format()方法来提供HTML样式格式。提供了以下信息显示：

- 生成特定的日志事件之前，从应用程序的开始所经过的时间
- 调用该记录请求的线程的名称
- 与此记录请求相关联的级别
- 日志记录器(Logger)和记录消息的名称
- 可选程序文件的位置信息，并从其中记录被调用的行号

HTMLLayout是一个非常简单的布局对象，它提供以下方法：

S.N.	方法 & 描述
1	**setContentType(String)** 设置 text/html 为 HTML内容的内容类型。默认为 text/html
2	**setLocationInfo(String)** 设置位置信息记录事件。默认为 false
3	**setTitle(String)** 设置为HTML文件的标题。默认值是Log4j的日志信息

HTMLLayout 例子：

以下是对HTMLLayout一个简单的配置文件：

```
# Define the root logger with appender file
log = /usr/home/log4j
log4j.rootLogger = DEBUG, FILE

# Define the file appender
log4j.appender.FILE=org.apache.log4j.FileAppender
log4j.appender.FILE.File=${log}/htmlLayout.html

# Define the layout for file appender
log4j.appender.FILE.layout=org.apache.log4j.HTMLLayout
log4j.appender.FILE.layout.Title=HTML Layout Example
log4j.appender.FILE.layout.LocationInfo=true
```

现在考虑下面的Java例子用于产生日志信息：

```
import org.apache.log4j.Logger;

import java.io.*;
import java.sql.SQLException;
import java.util.*;

public class log4jExample{
  /* Get actual class name to be printed on */
  static Logger log = Logger.getLogger(
```

```
                        log4jExample.class.getName());

  public static void main(String[] args)
                throws IOException,SQLException{

     log.debug("Hello this is an debug message");
     log.info("Hello this is an info message");
  }
}
```

编译并运行上述程序，它会在 /usr/home/log4j 目录创建 htmlLayout.html 文件，该文件将有如下的日志信息：

Log session start time Mon Mar 22 13:30:24 AST 2014

Time	Thread	Level	Category	File:Line	Message
0	main	DEBUG	log4jExample	log4jExample.java:15	Hello this is an debug message
6	main	INFO	log4jExample	log4jExample.java:16	Hello this is an info message

可以使用一个Web浏览器打开htmlLayout.html 文件。同样重要的是要注意，页脚</ HTML>和</ body>标记是完全缺失。具有HTML格式的日志文件的一大优势是，它可以被发布为网页可以远程查看。

2、PatternLayout

如果想生成基于模式的特定格式的日志信息，那么可以使用 org.apache.log4j.PatternLayout 格式化日志信息。PatternLayout类扩展抽象 org.apache.log4j.Layout 类并覆盖format()方法根据提供的模式构建日志信息。 PatternLayout也是一个简单的布局对象，它提供下列Bean属性，可以通过配置文件进行设置：

S.N.	属性和说明
1	**conversionPattern** 设置转换模式。默认为 %r [%t] %p %c %x - %m%n

模式转换字符：

下表说明了以上模式使用的字符和所有其他字符，可以在自定义模式中使用：

转换字符	表示的意思
c	用于输出的记录事件的类别。例如，对于类别名称"a.b.c" 模式 %c{2} 会输出 "b.c"
C	用于输出呼叫者发出日志请求的完全限定类名。例如，对于类名 "org.apache.xyz.SomeClass", 模式 %C{1} 会输出 "SomeClass".
d	用于输出的记录事件的日期。例如， %d{HH:mm:ss,SSS} 或 %d{dd MMM yyyy HH:mm:ss,SSS}.
F	用于输出被发出日志记录请求，其中的文件名

F	用于输出被发出日志记录请求，其中的文件名
l	用于将产生的日志事件调用者输出位置信息
L	用于输出从被发出日志记录请求的行号
m	用于输出使用日志事件相关联的应用程序提供的消息
M	用于输出发出日志请求所在的方法名称
n	输出平台相关的行分隔符或文字
p	用于输出的记录事件的优先级
r	用于输出毫秒从布局的结构经过直到创建日志记录事件的数目
t	用于输出生成的日志记录事件的线程的名称
x	用于与产生该日志事件的线程相关联输出的NDC（嵌套诊断上下文）
X	在X转换字符后面是键为的MDC。例如 X{clientIP} 将打印存储在MDC对键clientIP的信息
%	文字百分号 %%将打印%标志

格式修饰符：

默认情况下，相关资料原样输出。然而，随着格式修饰符的帮助下，可以改变最小字段宽度，最大字段宽度和对齐。

下表涵盖了各种各样的修饰符的情况：

Format modifier	left justify	minimum width	maximum width	注释
%20c	false	20	none	用空格左垫，如果类别名称少于20个字符长
%-20c	true	20	none	用空格右垫，如果类别名称少于20个字符长
%.30c	NA	none	30	从开始截断，如果类别名称超过30个字符长
%20.30c	false	20	30	用空格左侧垫，如果类别名称短于20个字符。但是，如果类别名称长度超过30个字符，那么从开始截断。
%-20.30c	true	20	30	用空格右侧垫，如果类别名称短于20个字符。但是，如果类别名称长度超过30个字符，那么从开始截断。

PatternLayout 示例：

以下是针对 PatternLayout 一个简单的配置文件：

```
# Define the root logger with appender file
log = /usr/home/log4j
log4j.rootLogger = DEBUG, FILE
```

```
# Define the file appender
log4j.appender.FILE=org.apache.log4j.FileAppender
log4j.appender.FILE.File=${log}/log.out

# Define the layout for file appender
log4j.appender.FILE.layout=org.apache.log4j.PatternLayout
log4j.appender.FILE.layout.ConversionPattern=
      %d{yyyy-MM-dd}-%t-%x-%-5p-%-10c:%m%n
```

现在考虑下面产生日志信息的Java例子：

```
import org.apache.log4j.Logger;

import java.io.*;
import java.sql.SQLException;
import java.util.*;

public class log4jExample{
  /* Get actual class name to be printed on */
  static Logger log = Logger.getLogger(
                      log4jExample.class.getName());

  public static void main(String[] args)
                throws IOException,SQLException{

     log.debug("Hello this is an debug message");
     log.info("Hello this is an info message");
  }
}
```

编译并运行上述程序，它会创建 log.out文件在 /usr/home/log4j 目录，该文件将有如下的日志信息：

```
2010-03-23-main--DEBUG-log4jExample:Hello this is an debug message
2010-03-23-main--INFO -log4jExample:Hello this is an info message
```