W6: Application Logging

.NET / Java

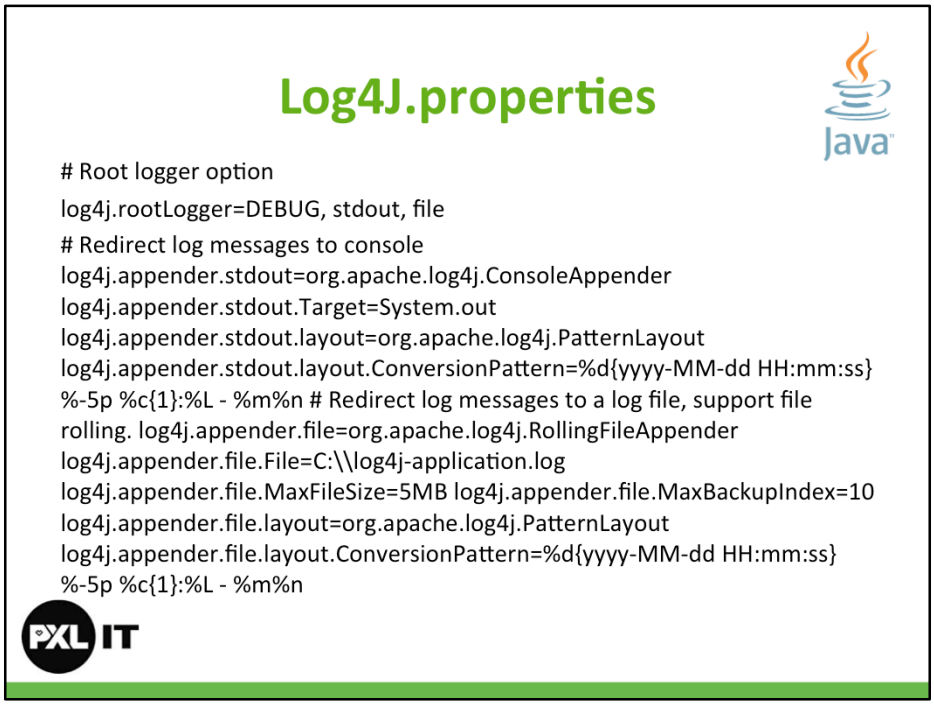
DE HOGESCHOOL
MET HET NETWERK

Hogeschool PXL – Dep. PXL-IT – Elfde-Liniestraat 26 – B-3500 Hasselt
www.pxl.be - www.pxl.be/facebook

Log4J

- Log4j.properties on classpath
- Get Log4j from MavenRepository

```
<dependency>
    <groupId>log4j</groupId>
    <artifactId>log4j</artifactId>
    <version>1.2.17</version>
</dependency>
```



Let break it down :

%d{yyyy-MM-dd HH:mm:ss} = Date and time format, refer to SimpleDateFormat JavaDoc.

%-5p = The logging priority, like DEBUG or ERROR. The -5 is optional, for the pretty print format.

%c{1} = The logging name we set via getLogger(), refer to log4j PatternLayout guide.

%L = The line number from where the logging request.

%m%n = The message to log and line break.

Note

For standalone Java app, make sure the log4j.properties file is under the project/classes directory

For Java web applications, make sure the log4j.properties file is under the WEB-INF/ classes directory

Log message examples

...
2014-07-02 20:52:39 DEBUG className:200 - This is debug message
2014-07-02 20:52:39 DEBUG className:201 - This is debug message2

How to log a Message?

- Declare logger:

final static Logger logger = Logger.getLogger(classname.class);

- Write code:

```
//logs a debug message
if(logger.isDebugEnabled()) {
        logger.debug("This is debug");
}
//logs an error message with parameter
logger.error("This is error : " + parameter);

//logs an exception thrown from somewhere
logger.error("This is error", exception);
```

Logger Priority

log4j.rootLogger=DEBUG, stdout

#...

Output:

2014-07-02 20:52:39 DEBUG HelloExample:19 - This is debug : PXL
2014-07-02 20:52:39 INFO HelloExample:23 - This is info : PXL
2014-07-02 20:52:39 WARN HelloExample:26 - This is warn : PXL
2014-07-02 20:52:39 ERROR HelloExample:27 - This is error : PXL
2014-07-02 20:52:39 FATAL HelloExample:28 - This is fatal : PXL


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

public class HelloExample {

        final static Logger logger = Logger.getLogger(HelloExample.class);

        public static void main(String[] args) {

                HelloExample obj = new HelloExample();
                obj.runMe(”PXL");

        }

        private void runMe(String parameter){

                if(logger.isDebugEnabled()){
```

Logger Priority

log4j.rootLogger=ERROR, stdout

#...

Output:

2014-07-02 20:52:39 ERROR HelloExample:27 - This is error : PXL
2014-07-02 20:52:39 FATAL HelloExample:28 - This is fatal : PXL

Logger Priority

If priority is defined in log4j.properties, only the same or above priority message will be logged.

```
package org.apache.log4j;
public class Priority {
  public final static int OFF_INT = Integer.MAX_VALUE;
  public final static int FATAL_INT = 50000;
  public final static int ERROR_INT = 40000;
  public final static int WARN_INT  = 30000;
  public final static int INFO_INT  = 20000;
  public final static int DEBUG_INT = 10000;
   //public final static int FINE_INT = DEBUG_INT;
  public final static int ALL_INT = Integer.MIN_VALUE;
```

How to log an Exception

```
try{
        obj.divide();
} catch(ArithmeticException ex) {
        logger.error("Sorry, something wrong!", ex);
}
```

Output:

2014-07-02 21:03:10 ERROR HelloExample2:16 - Sorry, something wrong! java.lang.ArithmeticException: / by zero at com.mkyong.HelloExample2.divide(HelloExample2.java:24) at com.mkyong.HelloExample2.main(HelloExample2.java:14)

Output to Console

- **log4j.properties:**

```
# Root logger option log4j.rootLogger=INFO, stdout
# Direct log messages to stdout
log4j.appender.stdout=org.apache.log4j.ConsoleAppender
log4j.appender.stdout.Target=System.out
log4j.appender.stdout.layout=org.apache.log4j.PatternLayout
log4j.appender.stdout.layout.ConversionPattern=%d{yyyy-MM-
dd HH:mm:ss} %-5p %c{1}:%L - %m%n
```

Output to File

- **log4j.properties:**

```
# Root logger option
log4j.rootLogger=INFO, file
# Direct log messages to a log file
log4j.appender.file=org.apache.log4j.RollingFileAppender
#Redirect to Tomcat logs folder
#log4j.appender.file.File=${catalina.home}/logs/logging.log
log4j.appender.file.File=C:\\logging.log
log4j.appender.file.MaxFileSize=10MB
log4j.appender.file.MaxBackupIndex=10
log4j.appender.file.layout=org.apache.log4j.PatternLayout
log4j.appender.file.layout.ConversionPattern=%d{yyyy-MM-dd
HH:mm:ss} %-5p %c{1}:%L - %m%n
```

Colored formatting

- **Zoek op internet hoe je kleuren kan gebruiken voor je logging patroon in Log4J.**

More info

- Manual:
  http://logging.apache.org/log4j/1.2/manual.html
- Pattern layout:
  http://logging.apache.org/log4j/1.2/apidocs/org/apache/log4j/PatternLayout.html