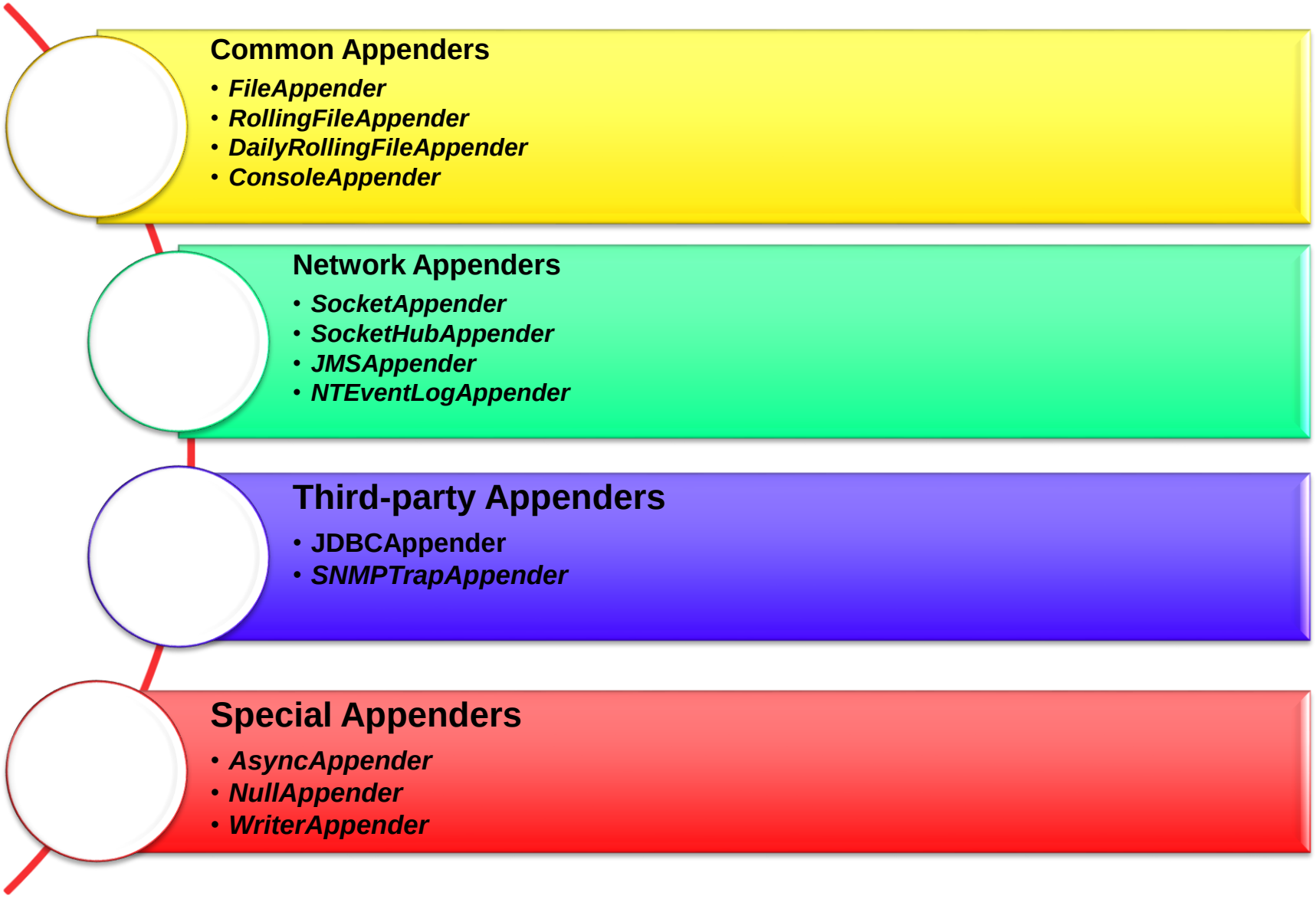
Appenders and Layout

Appenders and Layouts

Appenders

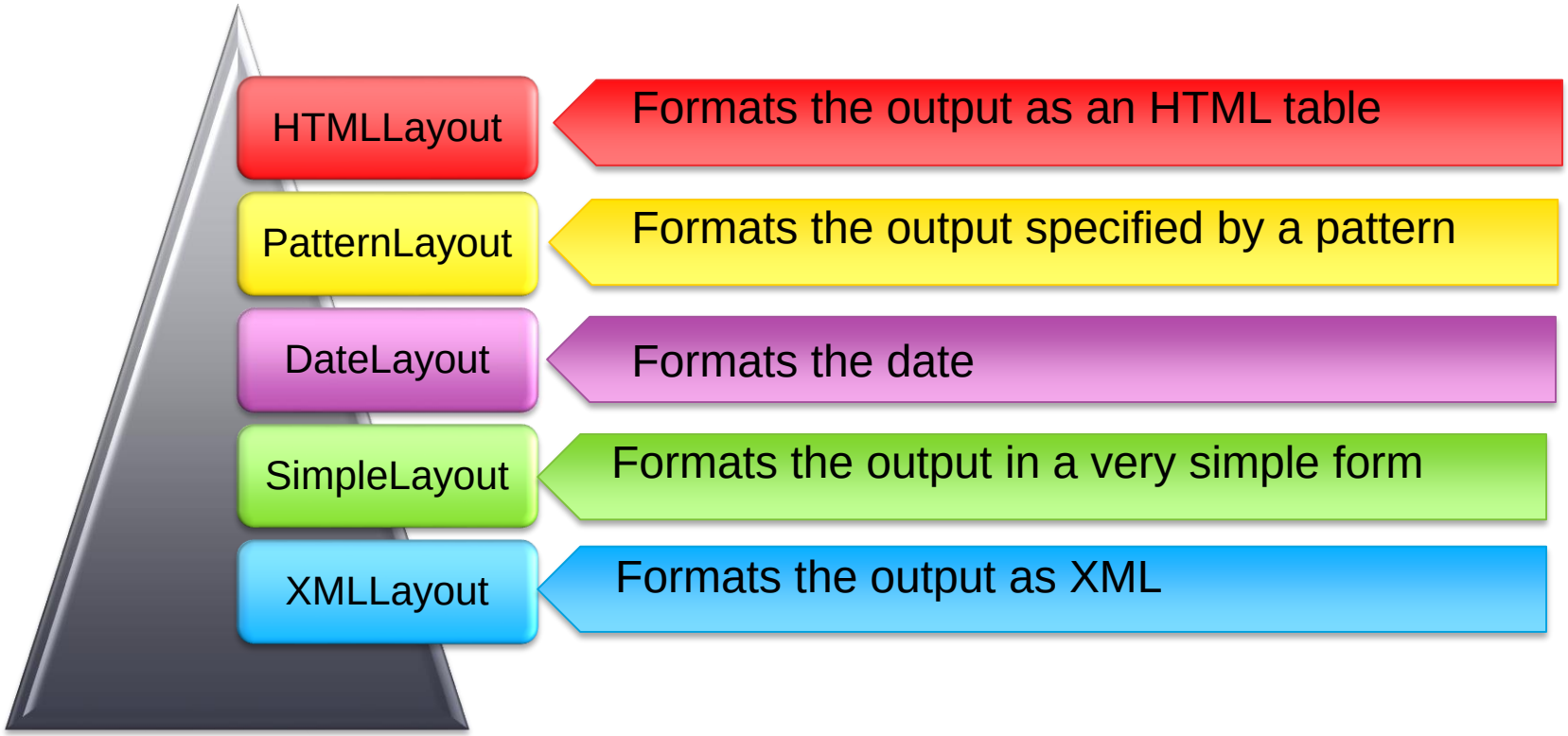
- In log4j *appender* publishes the output synchronously and asynchronously
- Currently, appenders are available for:
  - console,
  - files,
  - GUI components,
  - remote socket servers,
  - JMS,
  - NT Event Loggers, and
  - remote UNIX Syslog daemons

Appenders



Layout

- log4j has various Layout objects which formats the logging data
- Layout objects can also be created to format according to the application requirement
- Some concrete subclasses of abstract Layout are:



Layout

- Some Commonly used methods of Layout class are
  - public abstract boolean ignoresThrowable()
  - public abstract String format(LoggingEvent event)
  - public String getFooter()
  - public String getHeader()
  - public String getContentType()

Layout

- HTMLLayout adds the LogEvents in table format and generates HTML Page
- PatternLayout uses conversion pattern string and returns the logEvent
- These patterns are similar to C printf method's format String
  - It starts with % followed by controls like field, justification and padding

Patten	Describtion
m msg message	application supplied message from the logging event
c	name of the logger publishing the event
d date	the date of the logging event
ex exception throwable	Throwable trace bound to the LoggingEvent
p level	Level of the logging event
highlight	Adds colors to the result

Configuration

- To Insert Log codes to the application we need to configure the logging
- Configuration file can be created using 2 methodologies
  - XML
  - Java Properties (with .properties extension)
- Few important properties to be set are
  - rootLogger
  - Level
  - Appender.file
  - Appender.layout
- Log4j uses Class.forName to load classes, thus log4j classes and the properties file will not in the scope of the same classloader
- Resulting in log4j unable find my properties file in a J2EE or WAR application

Configuration file using 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=d:\\logigng.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
```

Specifying the root logger

Specifying the type of appender

Specifying the location of the log file

Specifying the Layout

Note:
In Properties file, comments start with # symbol

Configuration file using XML

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration xmlns:log4j="http://jakarta.apache.org/log4j/">
 <appender name="A1" class="org.apache.log4j.ConsoleAppender">
   <layout class="org.apache.log4j.PatternLayout">
     <!-- Print the date in ISO 8601 format -->
     <param name="ConversionPattern" value="%d %p [%t] %c - %m%n"/>
   </layout>
  </appender>
  <root>
   <priority value ="debug" />
   <appender-ref ref="A1" />
  </root>
</log4j:configuration>
```

Specifying the type of appender

Specifying the Layout

Writing to a log file Demo

- Create a Java Project
- Add log4j-1.2.17.jar to the project using Project->Build Path -> Add externals jars
- Create log.properties file in src folder of the project

log.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=d:\\logigng.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
```

Writing to a log file Demo

- Create a SimpleDemo class

SimpleDemo.java

```
package com.wipro.log4j;


import org.apache.log4j.Logger;

public class SimpleDemo {

static final Logger logger = Logger.getRootLogger();
public static void main(String[] args) {
logger.debug("Sample debug message");
logger.info("Sample info message");
logger.warn("Sample warn message");
logger.error("Sample error message");
logger.fatal("Sample fatal message");
}

}
```

Writing to a log file Demo

- Output in the log file:

2015-05-15 16:28:34 DEBUG root:10 - Sample debug message
2015-05-15 16:28:34 INFO root:11 - Sample info message
2015-05-15 16:28:34 WARN root:12 - Sample warn message
2015-05-15 16:28:34 ERROR root:13 - Sample error message
2015-05-15 16:28:34 FATAL root:14 - Sample fatal message

Log Demo Using XML Configuration

Simple Demo

- Create a Java Project
- Add log4j-1.2.17.jar to the project using Project->Build Path -> Add externals jars
- Create configuration.xml file under src folder of the project

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE log4j:configuration SYSTEM "log4j.dtd">
<log4j:configuration xmlns:log4j="http://jakarta.apache.org/log4j/">
 <appender name="A1" class="org.apache.log4j.ConsoleAppender">
    <layout class="org.apache.log4j.PatternLayout">
      <!-- Print the date in ISO 8601 format -->
      <param name="ConversionPattern" value="%d %p [%t]  %c - %m%n"/>
    </layout>
  </appender>
  <root>
    <priority value ="debug" />
    <appender-ref ref="A1" />
  </root>
</log4j:configuration>
```

configuration.xml

Simple Demo

- Create a SimpleDemo class

SimpleDemo.java

```
import org.apache.log4j.Level;
import org.apache.log4j.Logger;
import org.apache.log4j.xml.DOMConfigurator;

public class SimpleDemo {

private static Logger log=Logger.getLogger(SimpleDemo.class);

public static void main(String[] args){
DOMConfigurator.configure("src\\configuration.xml");
log.setLevel(Level.INFO);
log.trace("Trace");
log.info("info");
log.debug("debug");
log.warn("warn");
log.error("error");
log.fatal("fatal");
System.out.println("sone");
}
}
```

Assignment

Summary

- Appenders
- Layouts
- Configuration

Thank You