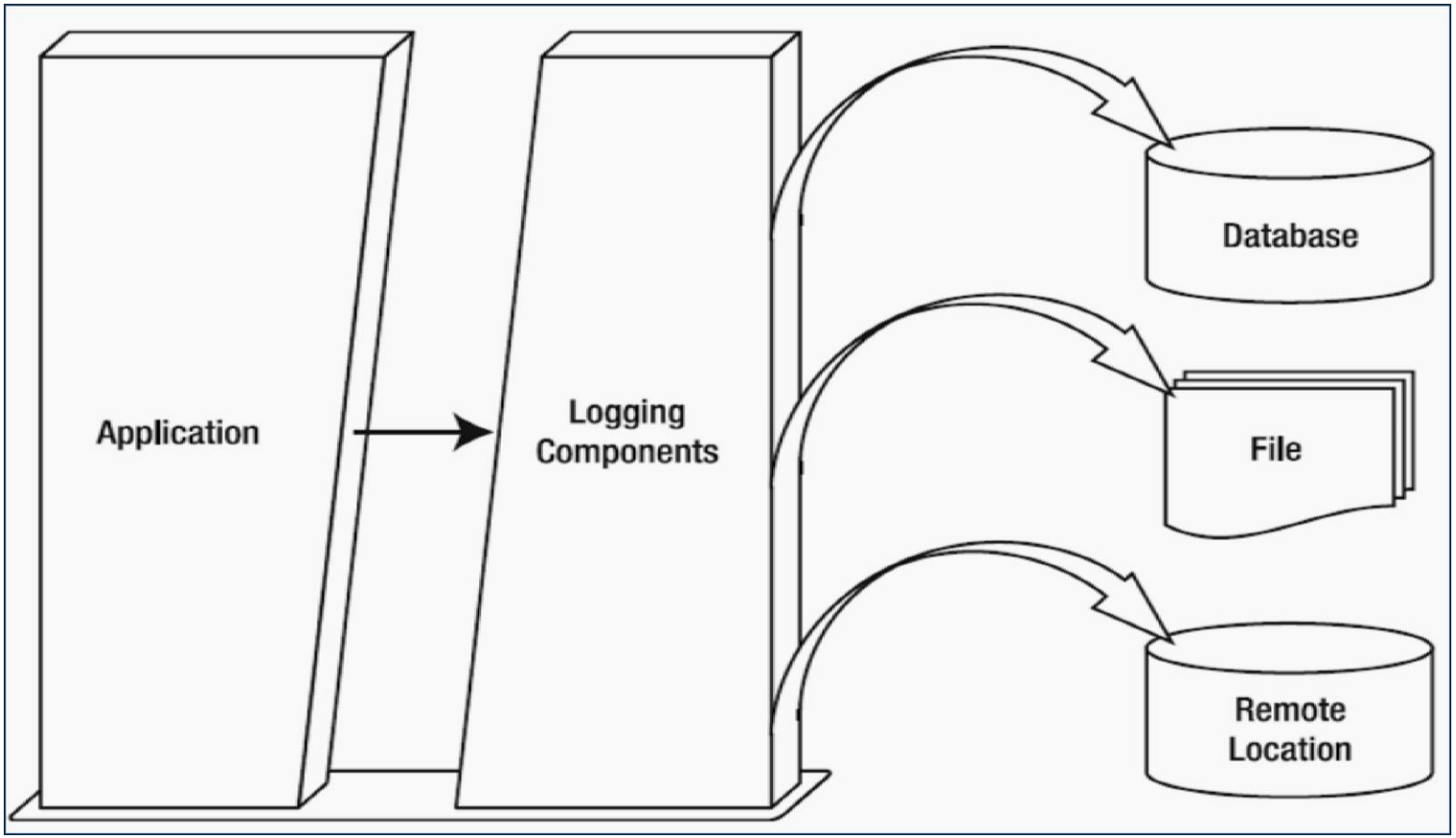
log4j

What is Logging

- Logging is a systematic and controlled way of representing the state of an application in a human-readable fashion.



log4j

- Apache log4j is an open-source logging API
- Log4j API is highly configurable through external configuration files at runtime
- Offers mechanisms to direct logging information to variety of destinations, such as –
  - Database
  - File
  - Console
  - Windows NT event log
  - UNIX Syslog
  - Java Message Service (JMS)

log4j Features

- log4j is thread-safe.
- log4j is optimized for speed.
- log4j is based on a named logger hierarchy.
- log4j supports multiple output appenders per logger.
- log4j supports internationalization.
- log4j is not restricted to a predefined set of facilities.
- Logging behavior can be set at runtime using a configuration file.
- log4j is designed to handle Java Exceptions from the start.
- log4j uses multiple levels, namely ALL, TRACE, DEBUG, INFO, WARN, ERROR and FATAL.
- The format of the log output can be easily changed by extending the *Layout* class.
- The target of the log output as well as the writing strategy can be altered by implementations of the Appender interface.
- log4j is fail-stop. However, altough it certainly strives to ensure delivery, log4j does not guarantee that each log statement will be delivered to its destination.

Log4j

Log4j has three main components:

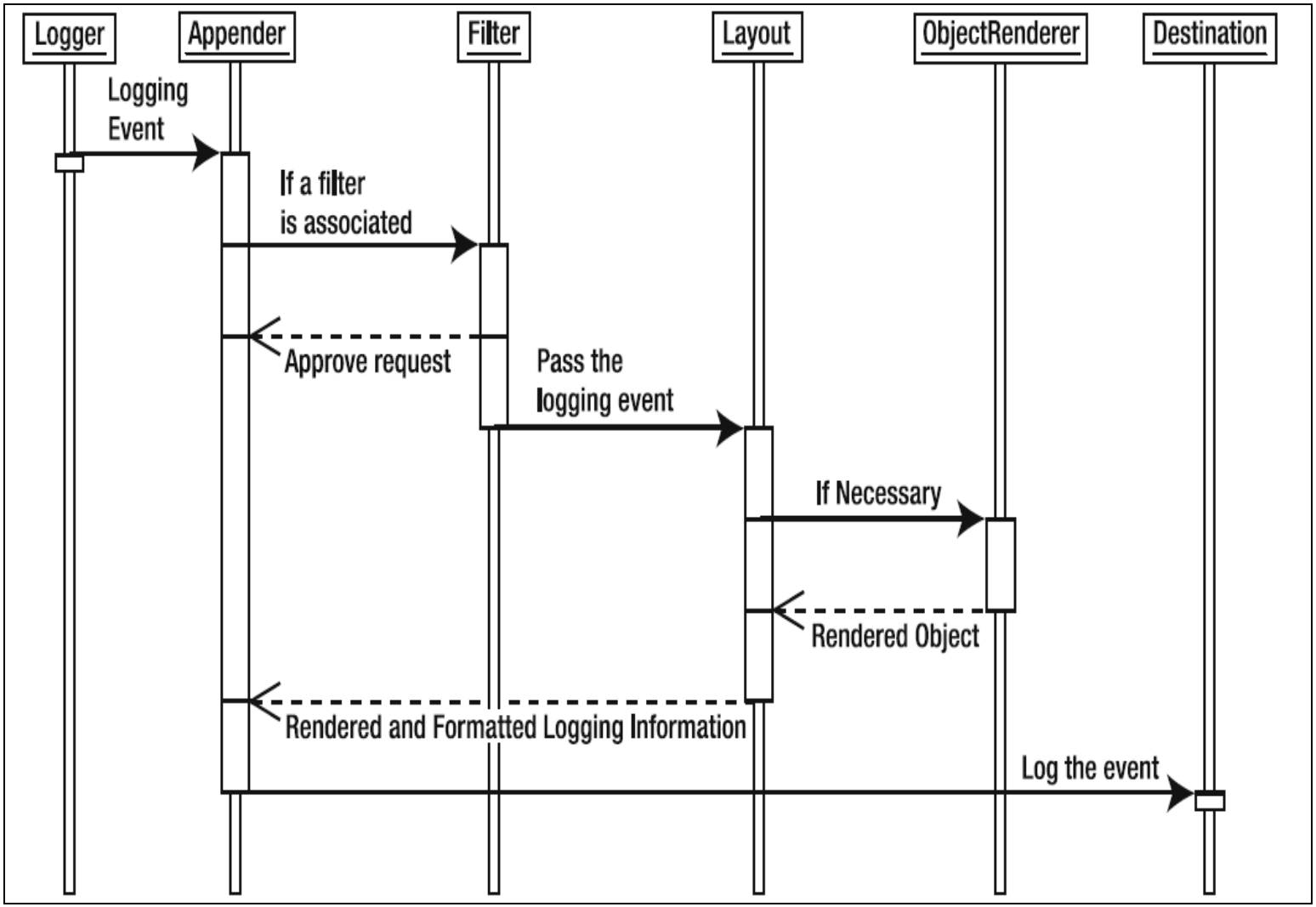
- Loggers
- Appenders
- Layouts

These three types of components work together to enable developers to log messages according to message type and level

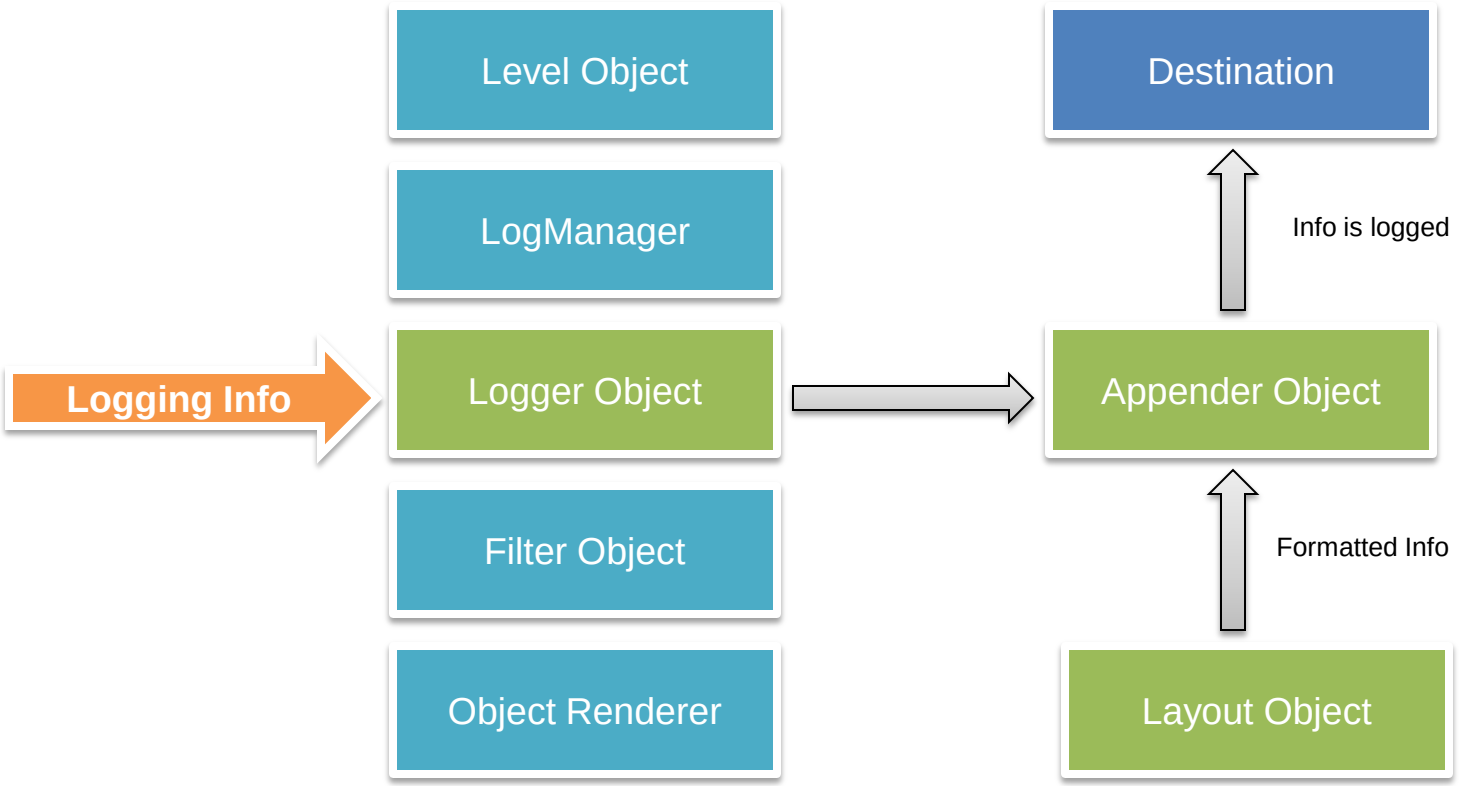
Log4j

- Logger:
  - The Logger object responsible for capturing logging information. Logger objects are stored in a namespace hierarchy.
- Appender:
  - The Appender object is responsible for publishing logging information to various preferred destinations. Each Appender object will have at least one target destination attached to it.
- Layout:
  - The Layout object is used to format logging information in different styles. Appender objects utilize Layout objects before publishing logging information. Layout objects play an important role in publishing logging information in a way that is human-readable and reusable

log4j Framework



log4j Framework



Loggers levels

The set of possible levels, that is:

Level	Description
ALL	All levels including custom levels.
DEBUG	Designates fine-grained informational events that are most useful to debug an application.
ERROR	Designates error events that might still allow the application to continue running.
FATAL	Designates very severe error events that will presumably lead the application to abort.
INFO	Designates informational messages that highlight the progress of the application at coarse-grained level.
OFF	The highest possible rank and is intended to turn off logging.
TRACE	Designates finer-grained informational events than the DEBUG.
WARN	Designates potentially harmful situations.

Sample Code

```
import org.apache.log4j.Logger;
 public class LogClass {
private static org.apache.log4j.Logger log = Logger
                                                .getLogger(LogClass.class);
    public static void main(String[] args) {
    log.trace("Trace Message!");
    log.debug("Debug Message!"); log.info("Info Message!");
    log.warn("Warn Message!");
    log.error("Error Message!");
    log.fatal("Fatal Message!");
}
}
```

Properties File

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

# Define the file appender
    log4j.appender.FILE=org.apache.log4j.FileAppender
    log4j.appender.FILE.File=log.out

# Define the layout for file appender
    log4j.appender.FILE.layout=org.apache.log4j.PatternLayout
    log4j.appender.FILE.layout.conversionPattern=%m%n
```

Thank You

Disclaimer

Tech Mahindra Limited, herein referred to as TechM provide a wide array of presentations and reports, with the contributions of various professionals. These presentations and reports are for informational purposes and private circulation only and do not constitute an offer to buy or sell any securities mentioned therein. They do not purport to be a complete description of the markets conditions or developments referred to in the material. While utmost care has been taken in preparing the above, we claim no responsibility for their accuracy. We shall not be liable for any direct or indirect losses arising from the use thereof and the viewers are requested to use the information contained herein at their own risk. These presentations and reports should not be reproduced, re-circulated, published in any media, website or otherwise, in any form or manner, in part or as a whole, without the express consent in writing of TechM or its subsidiaries. Any unauthorized use, disclosure or public dissemination of information contained herein is prohibited. Unless specifically noted, TechM is not responsible for the content of these presentations and/or the opinions of the presenters. Individual situations and local practices and standards may vary, so viewers and others utilizing information contained within a presentation are free to adopt differing standards and approaches as they see fit. You may not repackage or sell the presentation. Products and names mentioned in materials or presentations are the property of their respective owners and the mention of them does not constitute an endorsement by TechM. Information contained in a presentation hosted or promoted by TechM is provided “as is” without warranty of any kind, either expressed or implied, including any warranty of merchantability or fitness for a particular purpose. TechM assumes no liability or responsibility for the contents of a presentation or the opinions expressed by the presenters. All expressions of opinion are subject to change without notice.