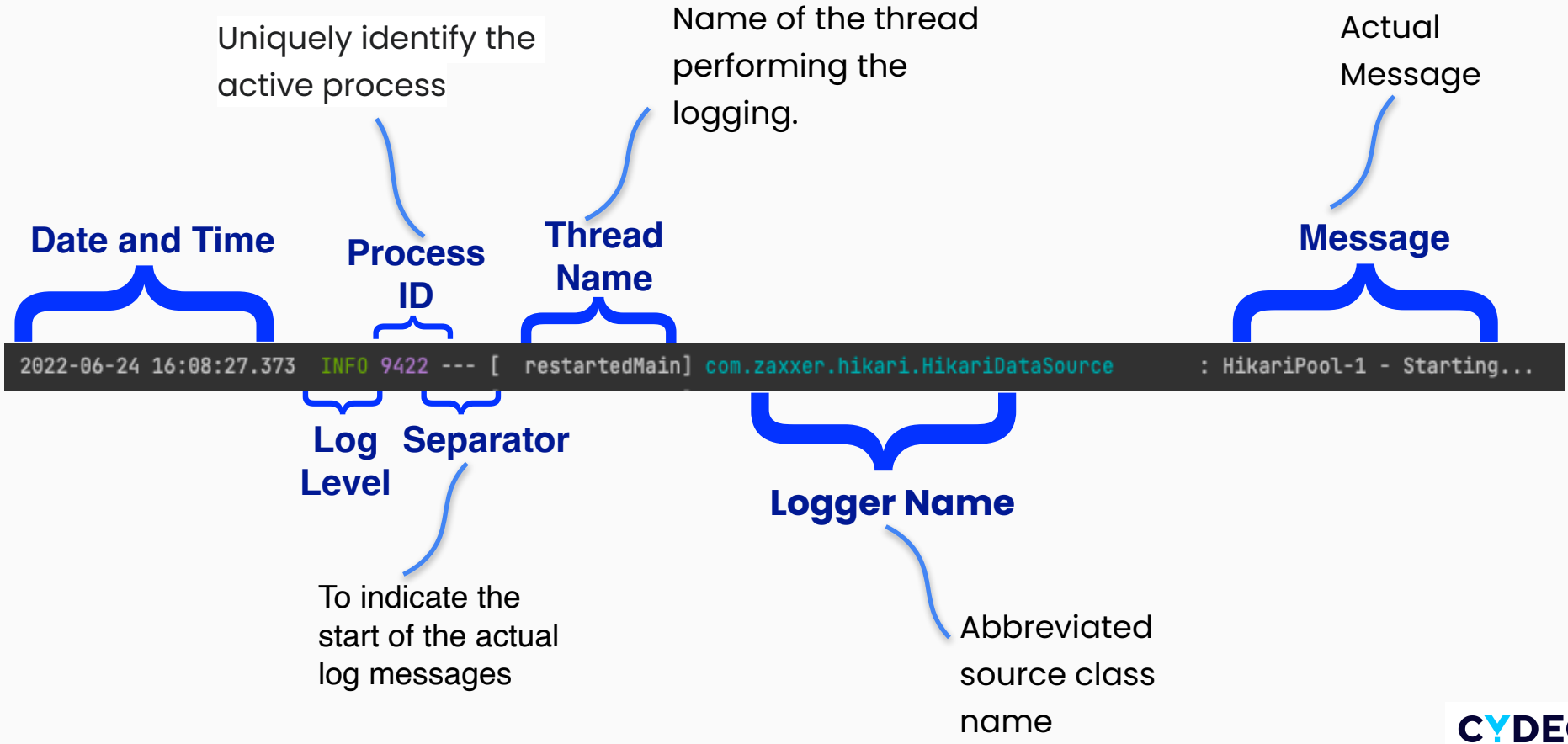
CYDEO

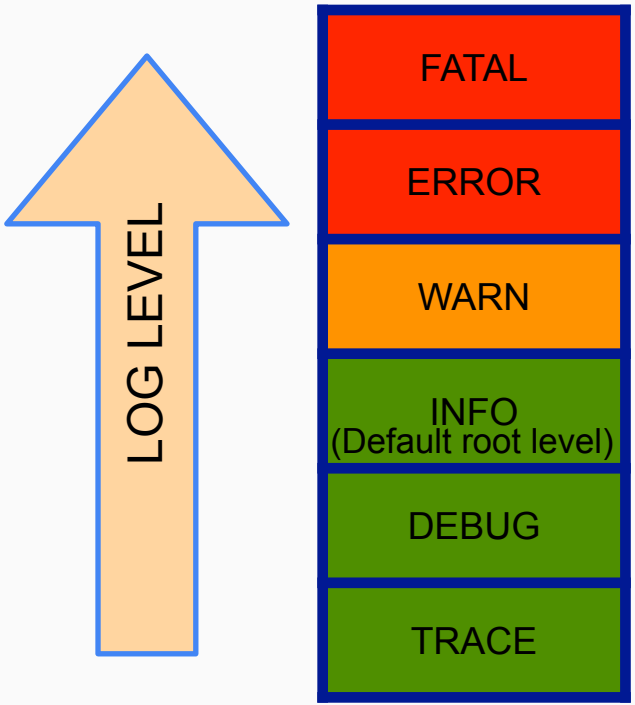
Logging

Log Statement

Log Statement



Log Level



- ✓ A logging level demonstrates the importance of the log statement.
- ✓ Any log statement logged with FATAL or ERROR indicates some serious issues in the application processing. Whereas INFO or DEBUG for example indicates typical regular application activities which you probably could ignore.

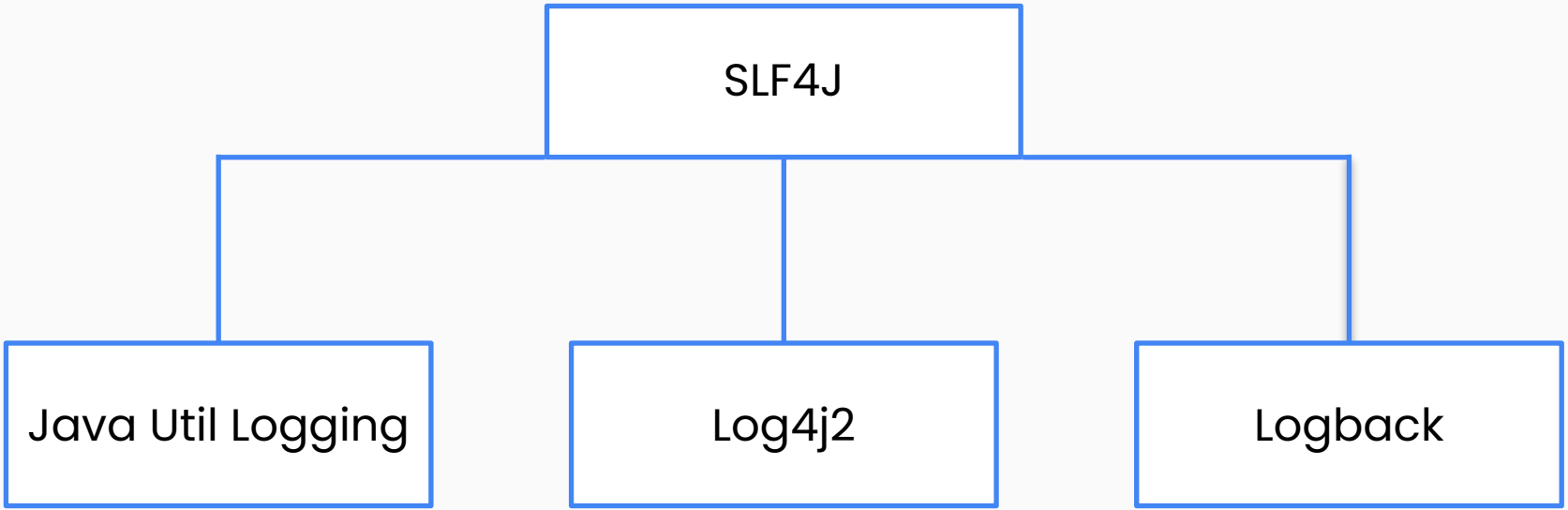
Pattern Layouts

- ✓ ***Date And Time*:** %d{yyyy-MM-dd HH:mm:ss.SSS}
- ✓ ***Log Level*:** %level
- ✓ ***Process ID*:** %pid or %processId
- ✓ ***Thread Name*:** %t
- ✓ ***Logger Name*:** %c or %c{1} - %c{2} - %c{3} ...
- ✓ ***Message*:** %msg
- ✓ ***Format Modifiers*:** %5level - %-6pid - %20.30c - %-20.-30c etc...

Storage for Logs

- ✓ Based on the logging configuration, log statements can be logged in various mediums such as in the console, files, and database. However, **console** and **file-based logging** are the dominant logging types and are most frequently used in an application.
- ✓ Although console logging works well in development time, in a production application, you need the log the application log statements in a file so that the file can be referred to in the future.
- ✓ There are the size and time-based policies to roll over the log file to a new file.

Log Frameworks



SLF4J

- The Simple Logging Facade for Java (SLF4J) serves as a simple facade or abstraction for various logging frameworks, such as java.util.logging, Logback and Log4j2.
- To be able to use Log4j2 in any Maven project, you will need to add this dependency:

```
<dependency>
    <groupId>org.apache.logging.log4j</groupId>
    <artifactId>log4j-core</artifactId>
    <version>2.17.2</version>
</dependency>
```

- If you want to use SLF4J+Log4j2 in any Maven project, you will need to add this dependency:

```
<dependency>
    <groupId>org.apache.logging.log4j</groupId>
    <artifactId>log4j-slf4j-impl</artifactId>
    <version>2.17.2</version>
</dependency>
```

Log4j2

- ✓ Log4j2 is the updated version of the popular and influential log4j library, used extensively throughout the Java ecosystem
- ✓ Log4j2 provides support for SLF4J, automatically reloads your logging configuration
- ✓ It allows for easy configuration of advanced logging best practices such as rolling files, different types of logging output destinations, support for structured logging formats

Log4j2 Architecture

- ***Configuration*:** The root element of a log4j2 configuration file; the *status* attribute represents the level at which internal log4j events should be logged
- ***Appenders*:** An appender in a logging framework primarily decides two main things – where the log messages should go, and what should be the logging format. Based on the destination of the log messages, there are several appender types.
- ***Loggers*:** A logger is a logging framework component that is responsible for logging the log messages using one or more appenders. You can define several loggers with various logging levels based on your need.

Configuring Appenders

- ***ConsoleAppender*** – logs messages to the *System* console
- ***FileAppender*** – writes log messages to a file
- ***RollingFileAppender*** – writes the messages to a rolling log file
- ***JDBCAppender*** – uses a relational database for logs
- ***AsyncAppender*** – contains a list of other appenders and determines the logs for these to be written in a separate thread
- **The RollingFileAppender** - writes to the File named in the fileName parameter and rolls the file over according the TriggeringPolicy and the RolloverPolicy
  - ***OnStartupTriggeringPolicy*** – a new log file is created every time the JVM starts
  - ***TimeBasedTriggeringPolicy*** – the log file is rolled based on a date/time pattern
  - ***SizeBasedTriggeringPolicy*** – the file is rolled when it reaches a certain size

Sources

- ***Log4j2***: https://logging.apache.org/log4j/2.x/
- ***Log4j2 Security Issues and Fixes***: https://logging.apache.org/log4j/2.x/security.html
- ***Log4j2 Manual Configuration***: https://logging.apache.org/log4j/2.x/manual/configuration.html
- ***Layouts / Pattern Layout***: https://logging.apache.org/log4j/2.x/manual/layouts.html