**Logging / Log4j2**

**Was ist Logging: (Alex)**

Logging ist die automatische Erstellung eines Protokolls (englisch log) von Softwareprozessen. Nützliche Protokolle können dem Entwickler (besonders wenn jemand den Code von den anderen debuggen/pflegen muss) eine enorme Hilfe sein, wenn er oder sie versucht zu verstehen, was der Code eigentlich tut.

Die Aufzeichnung kann beispielsweise auf der Kommandozeile, in einer Logdatei, einer spezialisierten Datenbank oder einem Webservice erfolgen. Meistens wird eine Logdatei benutzt.

**Logging frameworks: (Alex)**

* Java.util.logging,
* Logback (In Spring Bootals default -> [**spring-boot-starter-web dependency**](https://docs.spring.io/spring-boot/docs/1.5.x/reference/html/howto-logging.html)),
* Log4j2.

**Was ist ein Logger? (Dilo)**

* Logger ist ein Objekt, welches die Logging-Aufgaben einer Klasse übernimmt. Es wird über seinen Namen identifiziert. Z.B. Logger log = Logger.getLogger(MyApplication.MyClass)
* Damit kann das Logging-Verhalten einer Klasse gesteuert werden

**Was ist Log4j2? (Dilo)**

* Framework zum Loggen von Anwendungsmeldungen in Java
* Log4j2 gilt als Vorreiter für andere Logging-Frameworks (auch in andere Programmiersprachen)
* Log4j2 hat eine erhebliche Performanceverbesserung gegenüber dem Vorgänger. Sie enthält asynchrone Logger und unterstützt multi APIs wie SLF4j, commons logging und java.util.logging
* Die Api von Log4j2 ist unabhängig von ihrer Implementierung

**Funktionsweise: (Moritz)**

* Anstatt auftretende Fehler und Infomeldungen auf der Standardausgabe auszugeben, dient log4j dazu, die Meldungen über sogenannte Logger an das gewählte Loggingsystem weiterzuleiten („Appender“)
* Neben der Auswahl des Loggingsystems wird gleichzeitig aufgrund der Wichtigkeit („Log-Level“) der Meldung entschieden, ob diese überhaupt weitergeleitet wird
* Vorteil: Entwickler muss sich beim Erstellen des Programms nur um die Wichtigkeit der Meldungen Gedanken machen
* Die Filterung und Art der Ausgabe kann zur Laufzeit konfiguriert werden

**Einstufung der Wichtigkeit (Log-Level): (Moritz)**

**ALL**

Alle Meldungen werden ungefiltert ausgegeben

**TRACE**

ausführlicheres Debugging, Kommentare

**DEBUG**

allgemeines Debugging (Auffinden von Fehlern)

**INFO**

allgemeine Informationen (Programm gestartet, Programm beendet, Verbindung zu Host Foo aufgebaut, Verarbeitung dauerte SoUndSoviel Sekunden …)

**WARN**

Auftreten einer unerwarteten Situation

**ERROR**

Fehler (Ausnahme wurde abgefangen. Bearbeitung wurde alternativ fortgesetzt)

**FATAL**

Kritischer Fehler, [Programmabbruch](https://de.wikipedia.org/wiki/Programmabbruch)

**OFF**

Logging ist deaktiviert

**Appender: (Dani)**

=> Mittels Appender kann das Ausgabeziel der erzeugten Loggingausgaben festgelegt werden

Die wichtigsten Appender-Arten:

* **ConsoleAppender:** Gibt auf Standardausgabe aus
* **FileAppender:** Schreibt in eine Datei
* **RollingFileAppender:** Beginnt eine neue Datei ab einer gewissen Größe
* **JDBCAppender:** Schreibt in eine [Datenbank](https://de.wikipedia.org/wiki/Datenbank)
* **NTEventLogAppender:** Schreibt ins Ereignisprotokoll des Windows-Systems
* **SMTPAppender:** Schickt bei gewissen Meldungen eine Mail

**Konfiguration: (Dani)**

3 Arten, log4j zu konfigurieren:

* Properties Datei
* XML Datei
  + Empfohlen, da die Konfiguration vom Code getrennt ist
  + Man kann umkonfigurieren, ohne Modifikation oder Neustart der Applikation
* Im Code

**BEISPIEL IN SPRING (ECLIPSE)**

