Logging

Nein, keine Bäume fällen

Übersicht

- Wofür brauchen wir Logging?
- Was sollte man loggen?
- Wo wird geloggt?
- Log-Levels
- log4j2

Wofür brauchen wir Logging?

- Protokollierung von Softwareprozessen
- Fehleranalyse
- Überwachung bei Betrieb
- Verbesserung der Software

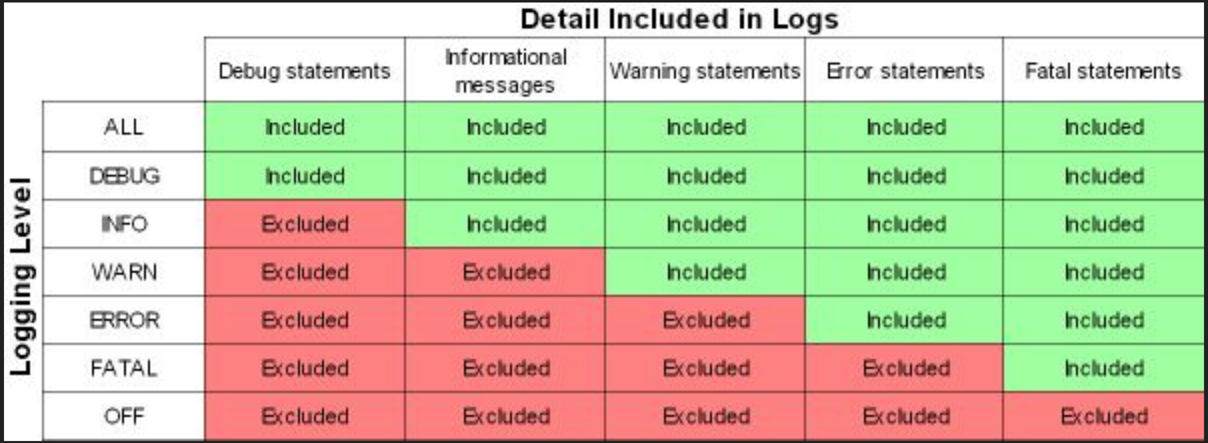
Was sollte man loggen?

- Schwerwiegende Fehler und Exceptions
- Applikations Events
- Debuginformationen
- Performance
- Abfragen
- Zeit und Dauer von Abläufen
- Threads

Wo wird geloggt?

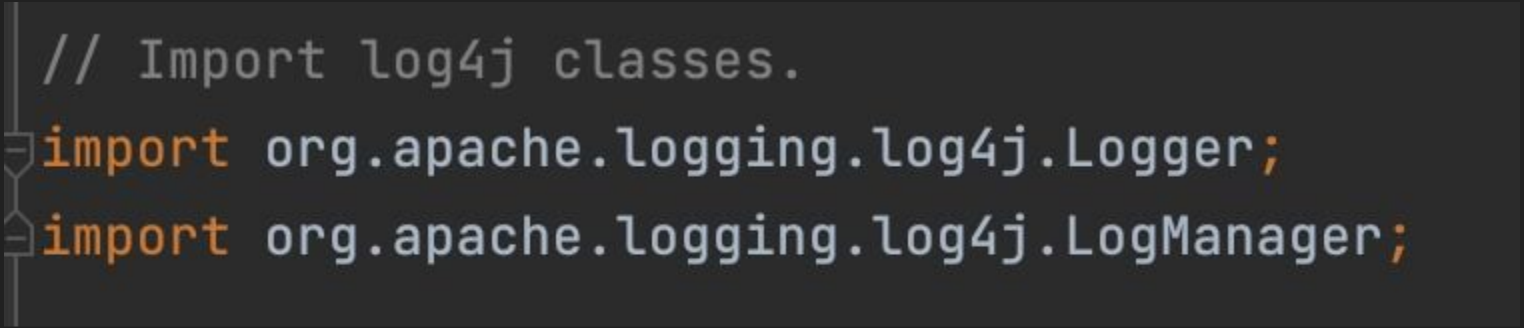
- Datei
- E-Mail
- Konsole
- Datenbank
- Web-Service

Log-Levels



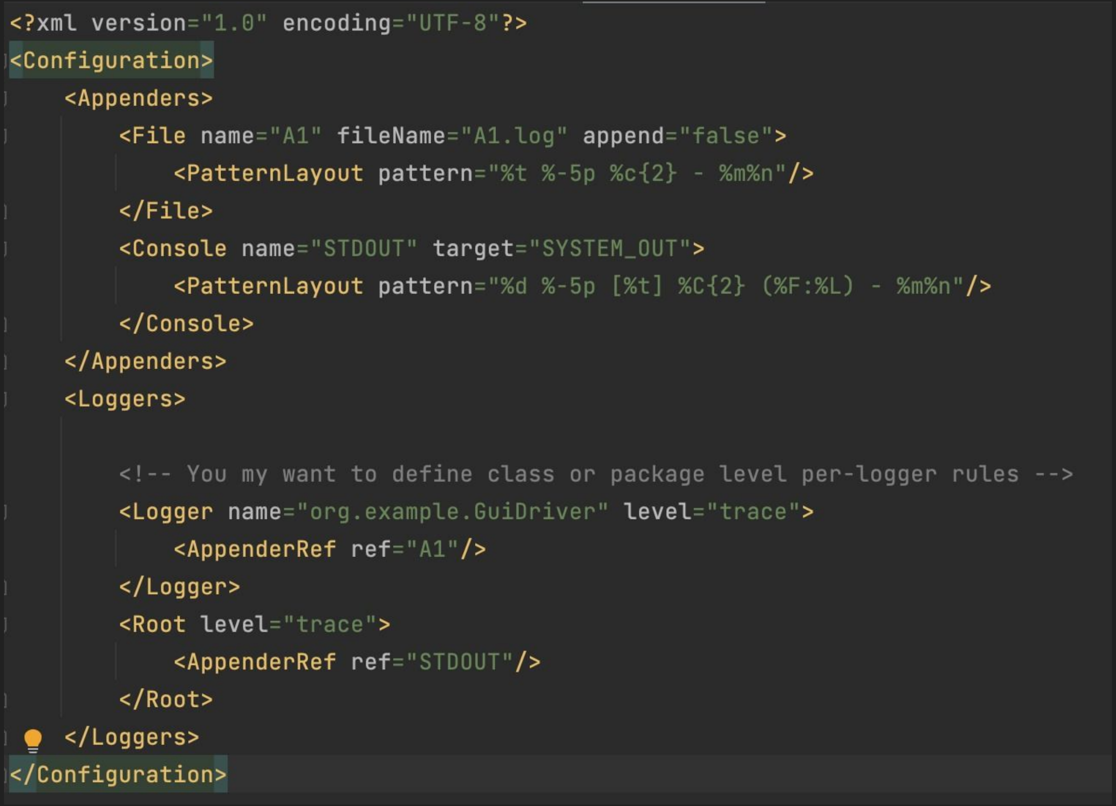
		Detail Included in Logs				
		Debug statements	Informational messages	Warning statements	Error statements	Fatal statements
Logging Level	ALL	Included	Included	Included	Included	Included
	DEBUG	Included	Included	Included	Included	Included
	INFO	Excluded	Included	Included	Included	Included
	WARN	Excluded	Excluded	Included	Included	Included
	ERROR	Excluded	Excluded	Excluded	Included	Included
	FATAL	Excluded	Excluded	Excluded	Excluded	Included
	OFF	Excluded	Excluded	Excluded	Excluded	Excluded

log4j2 Import



```java
// Import log4j classes.
import org.apache.logging.log4j.Logger;
import org.apache.logging.log4j.LogManager;
```

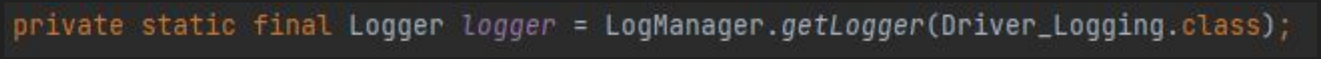
log4j2 Config



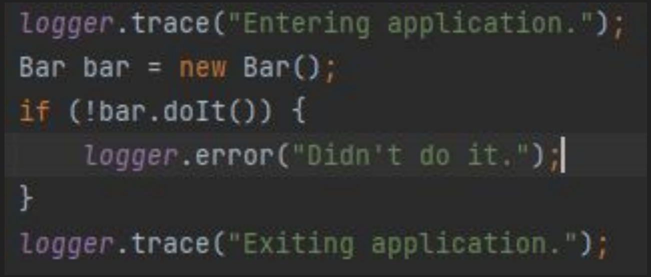
```xml
<?xml version="1.0" encoding="UTF-8"?>
<Configuration>
    <Appenders>
        <File name="A1" fileName="A1.log" append="false">
            <PatternLayout pattern="%t %-5p %c{2} - %m%n"/>
        </File>
        <Console name="STDOUT" target="SYSTEM_OUT">
            <PatternLayout pattern="%d %-5p [%t] %C{2} (%F:%L) - %m%n"/>
        </Console>
    </Appenders>
    <Loggers>

        <!-- You my want to define class or package level per-logger rules -->
        <Logger name="org.example.GuiDriver" level="trace">
            <AppenderRef ref="A1"/>
        </Logger>
        <Root level="trace">
            <AppenderRef ref="STDOUT"/>
        </Root>
    </Loggers>
</Configuration>
```

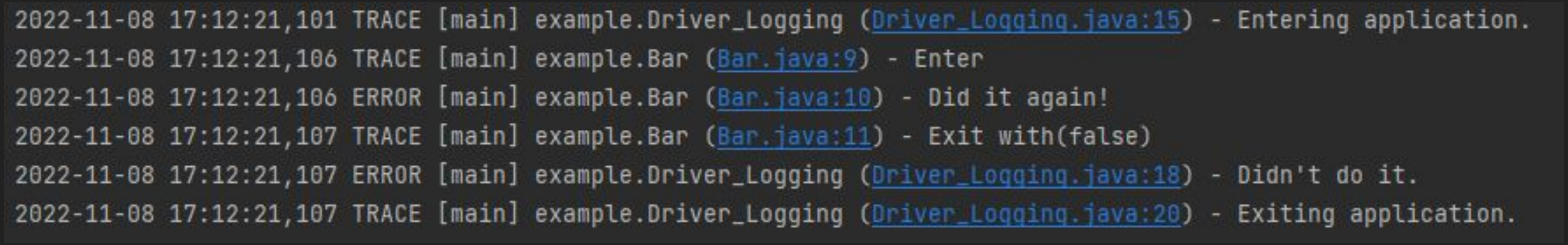
log4j2 Initialisierung und Verwendung



```java
private static final Logger logger = LogManager.getLogger(Driver_Logging.class);
```



```java
logger.trace("Entering application.");
Bar bar = new Bar();
if (!bar.doIt()) {
    logger.error("Didn't do it.");
}
logger.trace("Exiting application.");
```



```
2022-11-08 17:12:21,101 TRACE [main] example.Driver_Logging (Driver_Logging.java:15) - Entering application.
2022-11-08 17:12:21,106 TRACE [main] example.Bar (Bar.java:9) - Enter
2022-11-08 17:12:21,106 ERROR [main] example.Bar (Bar.java:10) - Did it again!
2022-11-08 17:12:21,107 TRACE [main] example.Bar (Bar.java:11) - Exit with(false)
2022-11-08 17:12:21,107 ERROR [main] example.Driver_Logging (Driver_Logging.java:18) - Didn't do it.
2022-11-08 17:12:21,107 TRACE [main] example.Driver_Logging (Driver_Logging.java:20) - Exiting application.
```

DANKE FÜR EURE AUFMERKSAMKEIT,
WENN IHR NOCH FRAGEN HABT,
DANN HÄTTET IHR AUFPASSEN
MÜSSEN
makeameme.org