Log4j 3.0

Matt Sicker, Software Engineer at Apple
PMC Member of Apache Logging Services

NOT A CONTRIBUTION

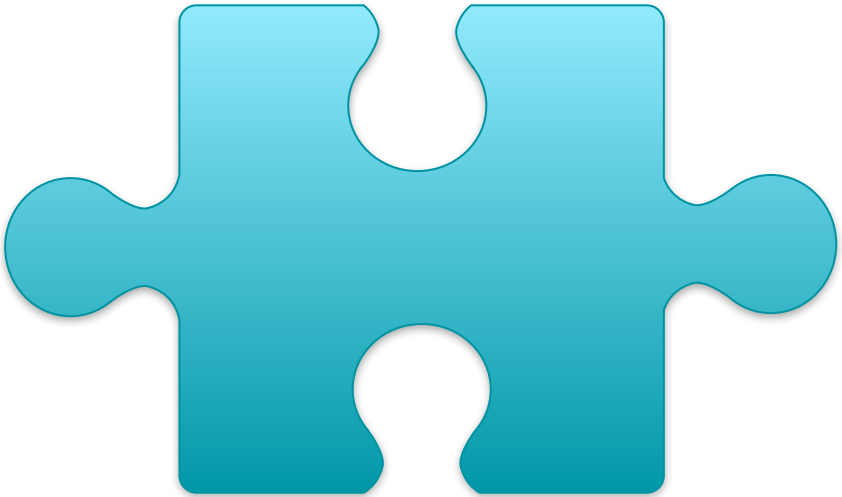
Agenda

- Modularization
- Plugins
- Configuration Properties
- Documentation
- Automation
- Q&A

Modularization

- Log4j 2 packaged largely in `log4j-core`, optional dependencies
- Log4j 3 packaged into several modules, required dependencies
  - Comparable to Spring Boot starters
  - Fine-grained opt-in feature flags
- Log4j API split for stable versioning
- Reduce attack surface of common usage
- Simplify dependency management
- Better integration with Java modules

Modularization

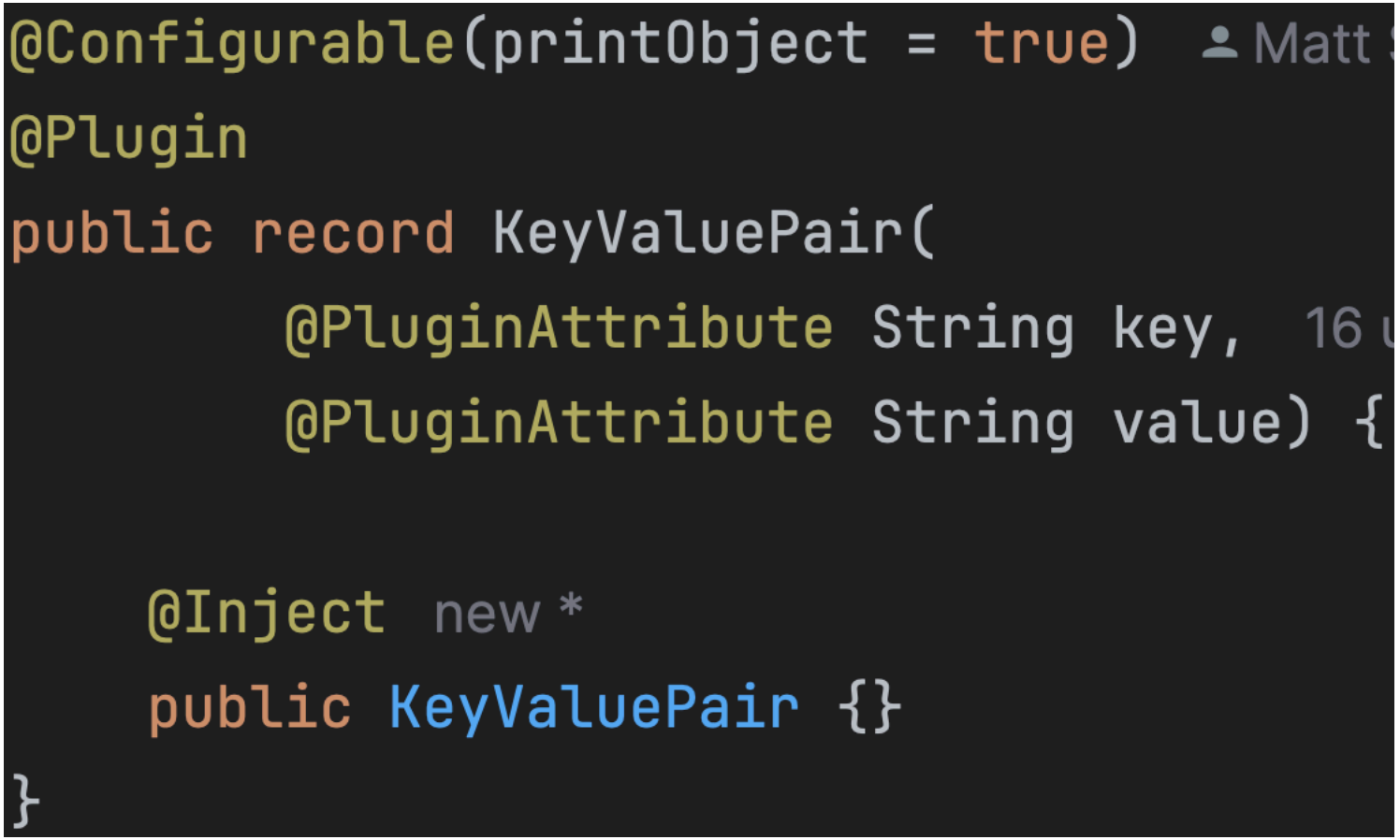


†Some assembly required

```
<module>log4j-async-logger</module>
<module>log4j-config-jackson</module>
<module>log4j-config-properties</module>
<module>log4j-config-yaml</module>
<module>log4j-conversant</module>
<module>log4j-core</module>
<module>log4j-core-test</module>
<module>log4j-csv</module>
<module>log4j-docker</module>
<module>log4j-gc-test</module>
<module>log4j-jctools</module>
<module>log4j-jdbc</module>
<module>log4j-jdbc-dbcp2</module>
<module>log4j-jdbc-jndi</module>
<module>log4j-jndi</module>
<module>log4j-jndi-test</module>
<module>log4j-jpl</module>
<module>log4j-jul</module>
<module>log4j-kit</module>
<module>log4j-layout-template-json</module>
<module>log4j-layout-template-json-test</module>
<module>log4j-mongodb</module>
<module>log4j-osgi-test</module>
<module>log4j-perf-test</module>
<module>log4j-plugin-processor</module>
<module>log4j-plugins</module>
<module>log4j-plugins-test</module>
<module>log4j-script</module>
<module>log4j-slf4j2-impl</module>
```

Plugins

- Annotations moved to `log4j-plugins`; annotation processor moved to `log4j-plugin-processor`
- Added `@Inject`-style dependency injection
- Custom components can be introduced via DI instead of system properties
  - 🆕 `Recycler` API for unified memory management
- Added `@Conditional` annotations for plugins and components
- Plugin metadata generated as `ServiceLoader`-loaded Java classes


@Configurable(printObject = true)
@Plugin
public record KeyValuePair(
@PluginAttribute String key,
@PluginAttribute String value) {
@Inject new *
public KeyValuePair {}
}

Configuration Properties

- Can be applied to individual `LoggerContext` instances or all
  - `log4j.`*`property-name`*
  - `log4j.contexts.`*`context-name.property-name`*
- Most properties split into components
  - Old example: `log4j2.loggerContextSelector`
  - New: `log4j.loggerContext.selector`

Documentation

- Documentation rewritten for Log4j 2.24.0 and 3.0.0
- Websites all standardized on AsciiDoc
- Documentation generated for all plugins
- XML schemas generated for all plugins
  - https://logging.apache.org/xml/ns
- Several new pages for beginners and high level overviews
- Various guides for common tasks

a subproject of **Apache Logging Services**

Getting started

This document aims to guide you through the most important aspects of logging with Log4j. It is not a comprehensive guide, but it should give you a good starting point.

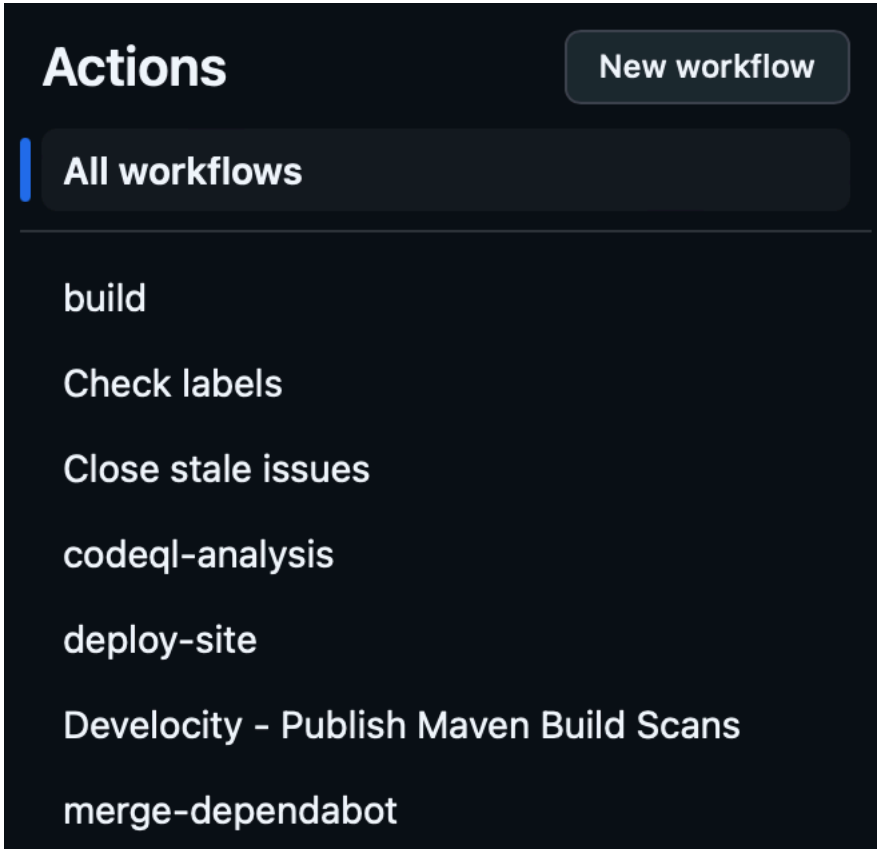
What is logging?

Logging is the act of publishing diagnostics information at certain points of a program execution. It means you can write messages to a log file or console to help you understand what your application is doing.

Contents

Automation

- Release process automated via GitHub Actions
- RCs created and signed at the push of a button
  - First in ASF history!
-  SBOMs  VDR
  - First in ASF history!
- Website publishing
- Plugin XML schemas publishing



When Will We Release 3.0.0?

- There are beta releases of 3.0.0 available
- Stable release of 3.0.0 expected in 2024
  - Java 17 baseline
- Continued support for 2.x releases
  - Java 8 baseline
- Other questions?
- https://musigma.blog/2023/11/10/log4shell-history.html
-