快看！发现一个新的小巧的日志组件 TingLog

> 废话不多说系列，直接开整
>
>
> ![cat001.png](https://p6-juejin.byteimg.com/tos-cn-i-k3u1fbpfcp/10b580685fa0449cae214fb065aae861~tplv-k3u1fbpfcp-jj-mark:3024:0:0:0:q75.awebp#?w=554&h=370&s=265284&e=png&b=8b8973)

一、介绍

常用日志框架：

1. Log4j 系列；
2. Logback 系列；
3. tinylog 轻量级的开源日志框架；

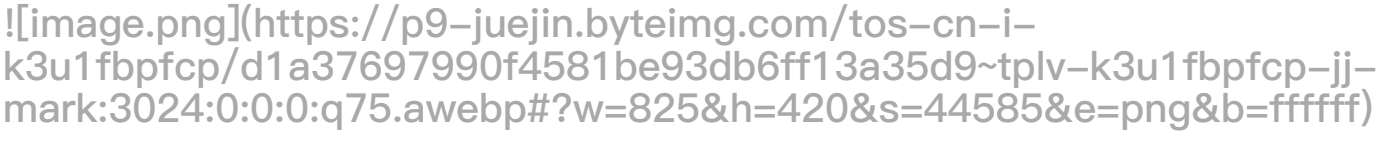
支持语言：

* Java（Java 6+）
* Kotlin
* Scala
* ...等其他 JVM 系语言（包括 Android）

支持日志级别：

* TRACE
* DEBUG
* INFO
* WARN
* ERROR
* OFF（禁用日志输出）

> 为什么是轻量级?
>
>
> * ① tiny log 主打轻量级，仅有两个 jar包，一个 API，一个是实现，无任何其他依赖；
> * ② 两个 jar 包总大小 不到 1M；
> * ③ 无需全局静态变量，开箱即用；



二、使用方式

（1）完整示例

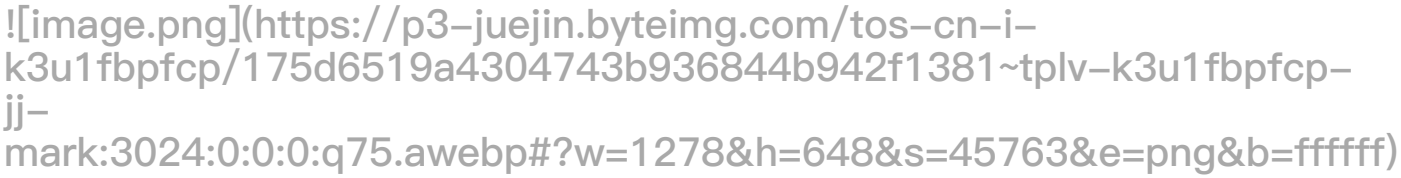
```
import org.tinylog.Logger;

public class Application {
    public static void main(String[] args) {
        Logger.info("hello world!");
    }
}
```

无需添加任何 Logger 这样的全局静态变量了：例如 Logback 需要使用静态变量到类中。直接类似使用自己定义的工具类那么使用。

```
// logback 日志框架：需要定义全局静态变量
private final Logger logger = LoggerFactory.getLogger(this.getClass());
```

（2）日志性能

基准测试报告：[tinylog.org/v2/benchmar...](http://cxyroad.com/"https://tinylog.org/v2/benchmark/")



（3）集成实战

① 引入依赖

```
<dependency>
    <groupId>org.tinylog</groupId>
    <artifactId>tinylog-impl</artifactId>
    <version>${tinylog.version}</version> <!-- 例如 2.4.1 -->
</dependency>
```

② 使用实例

```
import org.tinylog.Logger;

public class Application {
    public static void main(String[] args) {
        Logger.info("Hello World!");
    }
}

// 启动项目，默认控制台输出
// 2022-01-19 14:52:32 [main] Application.main()
// INFO: Hello World!
```

③ 简单配置日志输出格式

在 application.properties 或者在 src/main/resources 目录下创建一个 tinylog.properties 配置文件，例如配置信息如下：

```
level              = INFO # 默认为 trace
# tinylog日志格式
writer1            = console # 控制台输出格式定义
writer1.format     = {date: HH:mm:ss.SSS} {class}.{method}() {level}:
{message}
# tinylog日志输出文件地址
writer2            = rolling file # 日志文件输出格式定义
writer2.file       = logs/{date: yyyy-MM-dd}/log_{count}.log
writer2.policies = startup, daily: 02:00
writer2.format     = {date: HH:mm:ss} [{thread}] {level}: {message}:
{message}

```

含义：此时日志级别为：info，创建了两个 writer，一个输出到控制台，一个写入文件，并每天按时进行日志切割归档。

三、常用知识详解

（1）日志输出格式

```
// 无参数输出
Logger.info("Hello World!");

// 有参数输出(java语言)
Logger.warn("Divide {} by {}", a, b);
// 有参数输出-数字格式化（java语言）
Logger.trace("Income: {0.00} EUR", amount);
// 有参数输出-条件判断（Java）
Logger.info("There {0#are no files|1#is one file|1<are {} files}", count);

// 输出对象
Logger.error(LocalDate.now());

```

（2）异常输出 + 文本

```
// ex : Exception ex 异常栈信息
Logger.error(ex);
Logger.error(ex, ”{} : This is error messages”, module_tag);
Logger.trace(ex, ”Cannot divide {} by {}”, a, b);
```

（3）懒记录日志

有时，必须为日志记录专门计算消息或参数。对于昂贵的计算，建议使用lambda表达式进行延迟日志记录，因为只有在实际输出日志条目时才会计算这些日志。

```
Logger.info(() -> compute());
// Lambda表达式也可以作为带有“{}”占位符的文本的参数传递:
Logger.debug(”Expensive computation: {}”, () -> compute());
```

（4）标记

Tinylog支持对日志条目进行分类的标记。例如，标记可以作为格式模式的一部分输出，或者用于将日志条目转发给不同的作者。

```
Logger.tag(”SYSTEM”).info(”Hello World!”);
// 如果一个标签被广泛使用，一个被标记的记录器的实例可以被持有:
TaggedLogger logger = Logger.tag(”SYSTEM”);
// 从tinylog 2.4开始，甚至可以创建一个带有多个标签的日志记录器，将每个日志条目发送给所有定义的标签:
TaggedLogger logger = Logger.tags(”FOO”, ”BAR”, ”BAZ”);
// 通常，静态记录器用于发出无标记的日志条目。然而，也可以获得一个工作方式与静态记录器完全相同的未标记记录器实例:
TaggedLogger logger = Logger.tag(null);
```

（5）上下文变量赋值

Tinylog有一个基于线程的上下文，可以用额外的值充实日志条目。上下文值可以作为格式模式的一部分输出。存储的值只对设置了值的线程及其子线程可见。

```
ThreadContext.put(”user”, name);
```

存储值在显式删除之前是存在的。因此，在将线程返回到线程池之前，应该清除线程上下文。对于没有被重用的线程来说，这是不必要的。

```
ThreadContext.clear();
```

附录

* 官方网址：[tinylog.org/](http://cxyroad.com/ ”https://tinylog.org/”);
* 开源地址：[github.com/tinylog-org...](http://cxyroad.com/ ”https://github.com/tinylog-org/tinylog”);
* API 文档：[tinylog.org/v2](http://cxyroad.com/ ”https://tinylog.org/v2”);
* 官方示例：[tinylog.org/v2/external...](http://cxyroad.com/ ”https://tinylog.org/v2/external-resources/”) ([GitHub - tinylog-org/tinylog-spring-boot-example: Example Spring Boot application for logging with tinylog](http://cxyroad.com/ ”https://github.com/tinylog-org/tinylog-spring-boot-example”))

> 至此，非常感谢阅读
>
>
> ![cat001.png](https://p6-juejin.byteimg.com/tos-cn-i-k3u1fbpfcp/10b580685fa0449cae214fb065aae861~tplv-k3u1fbpfcp-jj-

mark:3024:0:0:0:q75.awebp#?w=554&h=370&s=265284&e=png&b=8b8973)

原文链接: https://juejin.cn/post/7362858912020201523