Difference between logger.info and logger.debug

Asked 11 years, 6 months ago Active 11 months ago Viewed 224k times

87

What is the difference between `logger.debug` and `logger.info` ?

When will `logger.debug` be printed?

java logging log4j

36

Share Edit Follow

edited Sep 25 '14 at 17:48
Oliver Lloyd
4,826 ● 6 ● 30 ● 55

asked Feb 26 '10 at 14:33
Senthilnathan
1,037 ● 2 ● 10 ● 9

8 Answers

Active | Oldest | Votes

123

I suggest you look at the article called "Short Introduction to log4j". It contains a short explanation of log levels and demonstrates how they can be used in practice. The basic idea of log levels is that you want to be able to configure how much detail the logs contain depending on the situation. For example, if you are trying to troubleshoot an issue, you would want the logs to be very verbose. In production, you might only want to see warnings and errors.

The log level for each component of your system is usually controlled through a parameter in a configuration file, so it's easy to change. Your code would contain various logging statements with different levels. When responding to an `Exception`, you might call `Logger.error`. If you want to print the value of a variable at any given point, you might call `Logger.debug`. This combination of a configurable logging level and logging statements within your program allow you full control over how your application will log its activity.

In the case of log4j at least, the ordering of log levels is:

```
DEBUG < INFO < WARN < ERROR < FATAL
```

Here is a short example from that article demonstrating how log levels work.

```
// get a logger instance named "com.foo"
Logger logger = Logger.getLogger("com.foo");

// Now set its level. Normally you do not need to set the
// level of a logger programmatically. This is usually done
// in configuration files.
logger.setLevel(Level.INFO);

Logger barlogger = Logger.getLogger("com.foo.Bar");
```

```
// This request is enabled, because WARN >= INFO.
logger.warn("Low fuel level.");

// This request is disabled, because DEBUG < INFO.
logger.debug("Starting search for nearest gas station.");

// The logger instance barlogger, named "com.foo.Bar",
// will inherit its level from the logger named
// "com.foo" Thus, the following request is enabled
// because INFO >= INFO.
barlogger.info("Located nearest gas station.");

// This request is disabled, because DEBUG < INFO.
barlogger.debug("Exiting gas station search");
```

Share Edit Follow

edited Feb 26 '10 at 14:49

answered Feb 26 '10 at 14:41

William Brendel
30.2k ●14 ●70 ●77

@momomo you might want to check this answer to know what `Trace` log level is. – RBT Feb 9 '17 at 8:10

Your example `If you want to print the value of a variable at any given point, you might call Logger.debug` helped me clarify the confusion I had between `Debug` and `Trace` level. Thank you! – RBT Feb 9 '17 at 8:11

34

This will depend on the logging configuration. The default value will depend on the framework being used. The idea is that later on by changing a configuration setting from INFO to DEBUG you will see a ton of more (or less if the other way around) lines printed without recompiling the whole application.

If you think which one to use then it boils down to thinking what you want to see on which level. For other levels for example in Log4J look at the API, http://logging.apache.org/log4j/1.2/apidocs/org/apache/log4j/Level.html

Share Edit Follow

answered Feb 26 '10 at 14:38

toomasr
4,623 ●2 ●29 ●36

16

Just a clarification about the set of all possible levels, that are:

```
ALL < TRACE < DEBUG < INFO < WARN < ERROR < FATAL < OFF
```

Share Edit Follow

answered Dec 5 '14 at 15:21

user1823890
515 ●7 ●6

11

Basically it depends on how your loggers are configured. Typically you'd have debug output written out during development but turned off in production - or possibly have *selected* debug categories writing out while debugging a particular area.

The point of having different priorities is to allow you to turn up/down the level of detail on a particular component in a reasonably fine-grained way - and only needing to change the logging configuration (rather than code) to see the difference.

Share Edit Follow

answered Feb 26 '10 at 14:36

Jon Skeet
1.3m ● 801 ● 8778 ● 8963

6

1. INFO is used to log the information your program is working as expected.
2. DEBUG is used to find the reason in case your program is not working as expected or an exception has occurred. it's in the interest of the developer.

Share Edit Follow

answered Sep 21 '20 at 13:39

Santosh Srivastava
61 ● 1 ● 1

4

This is a very old question, but i don't see my understanding here so I will add my 2 cents:

Every level corresponds/maps to a type of user:

- debug : developer - manual debugging
- trace : automated logging and step tracer - for 3rd level support
- info : technician / support level 1 /2
- warn : technician / user error : automated alert / support level 1
- critical/fatal : depends on your setup - local IT

Share Edit Follow

answered Jul 14 '19 at 11:44

Mickey Perlstein
1,994 ● 2 ● 24 ● 32

1

It depends on which level you selected in your log4j configuration file.

```
<Loggers>
        <Root level="info">
        ...
```

If your level is "info" (by default), `logger.debug(...)` will not be printed in your console. However, if your level is "debug", it will.

Depending on the criticality level of your code, you should use the most accurate level among the following ones :

```
ALL < TRACE < DEBUG < INFO < WARN < ERROR < FATAL < OFF
```

Share Edit Follow

answered Nov 15 '17 at 13:06

Jack'
903 ● 11 ● 20

0

What is the difference between logger.debug and logger.info?

These are only some default level already defined. You can define your own levels if you like. The purpose of those levels is to enable/disable one or more of them, without making any change in your code.

When logger.debug will be printed ??

When you have enabled the debug or any higher level in your configuration.

Share Edit Follow

answered Feb 26 '10 at 14:44

craftsman
14k ● 17 ● 62 ● 82