Introduction to logging

Theory Practice 100% completed, 0 problems solved

Theory

7 minutes reading

Unskip this topic Start practicing

1 required topic

Computer programming

8 dependent topics

Debugging techniques

Standard logger

Debugging techniques

Errors

Intro to SQL Alchemy

Standard logging for JVM

Logging

Introduction to logging in Java

Table of contents:

§1. Introduction

Imagine you are investigating a robbery and are trying to figure out what happened. You have witnesses whose testimonies are very vague, but there is no evidence. It will be very difficult to unravel such a mess. If only there were recordings from surveillance cameras to restore the chronology of the events...

So now, let's say you are investigating a bug, but what 'surveillance cameras' do you have in that case? Obviously, we need records of everything that happened with the program before the incident. Also, it would be nice to know during which operation it happened. All such records are usually kept in a log. So what is a log?

§2. Log

Logging refers to the act of recording during the execution of an application. **Logs** are records that give us information about the events in a software application. This record could be a message that is enough to understand the event that happened, it may include a timestamp, a description, and a severity level. These events could be user-generated or system-generated. We use either a log file or, sometimes, the standard output to make these records.

We need logging for several reasons. Firstly, it will save a lot of time when we are **troubleshooting** our application at a late stage. If the program, for example, broke or something went wrong in it, then we can find the exact moment at which the error occurred in the log. This makes the debugging process a lot easier. Secondly, it is also possible to trace who used the program with the log. If it is, say, a site, one can find out who sent the requests. Also, logs help to monitor the operation of a particular system, which makes verification and reporting a whole lot easier. This way, we are always aware of how our programs work and how well they perform.

There are several things to consider when logging: *when do we log? what do we log? how do we log?* Let's find out the answers to these questions.

§3. When, What, and How

As we've mentioned above, there are several common reasons to generate logs:

- troubleshooting
- auditing
- profiling
- statistics

What we log usually depends on the application. In any case, we should be able to understand the execution path of a program through the logs. It is important to avoid excessive logging as it is costly. For example, there's no need to log the start and the end of every method, their arguments, since they are easy to track. Logs are meant to cover server issues, database issues, networking issues, errors from unanticipated user inputs, states of dynamically created objects, configuration values.

Providing contextual information in your log messages is very important as well. Often, the success or the failure of a program depends on the user inputs. So, you need to put them in your log messages if necessary. For example, when authenticating a user, log the username that is inputted. Context is also important when your program runs in a concurrent environment. In such a case the thread name can be added to the log message.

§4. Log levels

We said earlier that a lot of important information can be added to the log file. But what kind of information is it? There are different types that correspond to the accepted logging levels: Debug, Info, Warn, Error, Fatal (from the least critical level to the most critical one).

Let's see what those log levels are for.

Debug logs are used to diagnose applications. Debugging messages inform us about what the program is doing at a certain step and what it gets as a result of these actions. For example, a message can contain information about the output of some function, so that we can see if it should be fixed.

Info is used to log important information about an application. It is used to log service start, service stop, configurations, assumptions. For example, the information about the user who has just registered on the website.

Warn logs are considered to be the first level of application failure. They are usually applied to log repeated attempts to access a resource, missing secondary data, or switching from a primary server to a back-up server. For instance, there can be a message about a possible disconnection with the server. However, it does not affect the user at this point.

Error log level is used for more critical problems. These kinds of issues usually affect the result of the operation but do not terminate the program. Errors are considered to be the second level of application failures. For example, there can be a message that a user could not log in because the database was temporarily unavailable.

Fatal is the third level of application failures. It is used to indicate a much more serious error that causes the termination of the program. Such a message may say that the program totally failed and depending on the time and conditions of this failure the developers can find out how to fix the problem.

Great, now we know what bugs are usually written into the log file. Now we need to display the log. But what should be displayed? In order to make the log readable and understandable, there is a special recording format. Let's find out more about it below.

§5. Log Format

To investigate a bug, we need to know when it happened, how serious it was, and who came across it. Thus, the log format generally looks like this:

```
[date time][log level][message]
```

So, it starts with the date and time of when the error occurred. Then comes the log level, and the last thing is the message with the explanation of what exactly happened. More specifically, if we, for example, want to monitor who registers on our site, we need the corresponding logs with **Info** log level. Then every time a user sends data to the site, we will log a message about this event. For example, on February 2, 2021, a user with the nickname 'demo' registered on the site at 3 pm. Then the log will look like this:

```
[2021-02-02 15:00:00] [INFO] User 'demo' has registered
```

And if some user named 'alex98' cannot log in because of some technical issues, we will receive an **Error** message:

```
[2021-02-02 01:00:10] [ERROR] User 'alex98' cannot log in because the
database is temporarily unavailable
```

Thus, we will know that user 'alex98' failed to log in due to our database being temporarily unavailable. We have localized the problem and know exactly what we need to do: check the database and fix it.

There is also a more complex version of this format called The Common Log Format.

§6. Conclusion

So, we've figured out what logs are and what they are for. We've also learned what types of logs there are and how we can issue a log message. Now you can start using logs to monitor the operation of your system at any given time.