16

Logging

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Objectives

After completing this lesson, you should be able to:

- Add a logging mechanism to your Java application
- Log messages to a file at an appropriate level
- Configure your logger
- Configure logging formatters and handlers

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Topics

- Logging API overview
- Logging configuration
- Logging examples
- Configuration files and custom handlers

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Logging Overview

For software applications, logging systems provide end users, system administrators, and developers with reports for troubleshooting and maintaining their systems. Logging systems are typically used for:

- Monitoring security
- Providing configuration information
- Reporting on errors or problems with a system
- Troubleshooting systems

ORACLE

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

When considering whether to use logging in you application, consider this statement from Brian W. Kernighan and Rob Pike from the book *The Practice of Programming*:

As personal choice, we tend not to use debuggers beyond getting a stack trace or the value of a variable or two. One reason for this is that it is easy to get lost in the details of complicated data structures and control flow; we find stepping through a program less productive than thinking harder and adding output statements and self-checking code at critical places. Clicking over statements takes longer than scanning the output of judiciously-placed displays. It takes less time to decide where to put print statements than to single-step to the critical section of code, even assuming that we know where that is. More important, debugging statements stay with the program; debugging sessions are transient.

Reference: http://logging.apache.org/log4j/1.2/manual.html

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Using the Java Logging API

To use the Java Logging API you need to:

- Create a logger object
  - Logging objects are found in the `java.util.logging` package

```
Logger logger = Logger.getLogger("com.example.ClassName");
```

- Configure the logger object
- Send messages to the logger object

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

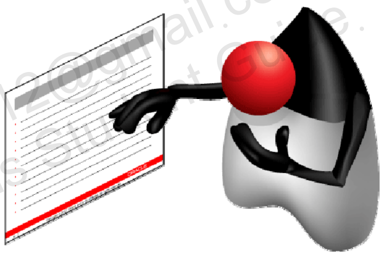
The Java Logging API classes can be found in the `java.util.logging` package.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Topics

- Logging API overview
- Logging configuration
- Logging examples
- Configuration files and custom handlers



Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Configure the Logger Handler

The logging API is contained in the `java.util.logging` package. The core class in the API is the **Logger** class. Setting the level for a Logger determines what messages are logged. A Logger object is associated with one or more handlers. The handler is responsible for publishing the log message. Predefined handlers include:

- `StreamHandler`
- `ConsoleHandlder`
- `FileHandler`
- `SocketHandler`
- `MemoryHandler`

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

ORACLE

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

A logger object is associated with one or more handlers. The handler is responsible for the publication of the log message. This responsibility includes specifying both the format and the output location. Examples of an output location are a console and a file.

The logging API provides the following handlers:

- `StreamHandler`: A simple handler for writing formatted records to an OutputStream object
- `ConsoleHandler`: A simple handler for writing formatted records to the System.err stream
- `FileHandler`: A handler that writes formatted log records either to a single file or to a set of rotating log files
- `SocketHandler`: A handler that writes formatted log records to remote TCP ports
- `MemoryHandler`: A handler that buffers log records in memory

Note: The default handler associated with the logger instance is the `StreamHandler`.

Configure the Logger Formatter

Once a handler is assigned, set the Formatter that determines the format of the log file. There are two predefined formatters:

- `SimpleFormatter`
- `XMLFormatter`

The `XMLFormatter` is the default in Java 7.

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Java includes two standard Formatters:

`SimpleFormatter`: Writes brief “human-readable” summaries of log records.

`XMLFormatter`: Writes detailed XML-structured information.

As with Handlers, it is fairly straightforward to develop new Formatters.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Configure the Logger Level

Logging output is controlled by using levels to define the importance of messages.

- The predefined levels, from highest to lowest, are:
  - SEVERE
  - WARNING
  - INFO
  - CONFIG
  - FINE
  - FINER
  - FINEST
- Two special levels of ALL or OFF are also available.

ORACLE

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

The level information is used in two ways: Firstly for recording logging error messages. SEVERE error would be the most serious, and probably should always be logged. FINEST would be the least important and probably should not be logged.

Setting the log lever for a logger. For example, `logger.setLevel(Level.INFO);` would log any messages that had their severity set to INFO, WARNING, or SEVERE.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Quiz

Setting the logging level:

a. Determines the location and the name of the log file
b. Has no effect on the log file
c. Provides military grade security and encryption to your application, making it much tougher to hack
d. Determines which messages are logged and, therefore, the amount of detail in the log

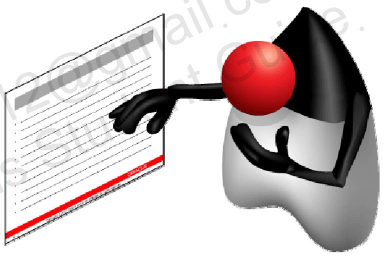
Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

ORACLE
Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Answer: d

Topics

- Logging API overview
- Logging configuration
- Logging examples
- Configuration files and custom handlers



Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Logger Example: `main()`

```
public static void main(String[] args) {
   BasicLogging bl = new BasicLogging();
   bl.logger = Logger.getLogger("com.example.BasicLogging");

   try {
       bl.logger.addHandler(new FileHandler("Basic.log"));
       bl.logger.setLevel(Level.INFO);
       bl.logMessages();
   } catch (IOException e){
       e.getMessage();
   }
}
```

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Logger Example: `logMessages()`

```
public void logMessages(){
    logger.severe("A very severe problem has occurred");
    logger.warning("Warning, something bad may be happening");
    logger.log(Level.INFO,"Here is INFO you may want to know
   about");
    logger.config("Here is some info about your CONFIG");
    logger.fine("This is some really FINE info");
    logger.finer("This is even FINER info");
    logger.finest("This is the FINEST info");
}
```

ORACLE
Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Logger Example: `basic.log`

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<!DOCTYPE log SYSTEM "logger.dtd">
<log>
<record>
  <date>2011-12-21T19:11:51</date>
  <millis>1324519911801</millis>
  <sequence>0</sequence>
  <logger>com.example.BasicLogging</logger>
  <level>SEVERE</level>
  <class>com.example.BasicLogging</class>
  <method>logMessages</method>
  <thread>1</thread>
  <message>A very severe problem has occured</message>
</record>
<!- Additional records here -->
</log>
```

ORACLE
Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

If you look at the file created from the program, only three messages are included the SEVERE, WARNING, and INFO messages. The other messages are not included, why?

Because of the logging level. Setting the logging level to INFO means that only messages of the INFO level and more severe are included.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.
Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Logger Example: `SimpleLogging.java`

```
public static void main(String[] args) {
    SimpleLogging sl = new SimpleLogging();
    sl.logger = Logger.getLogger("com.example.SimpleLogging");

    try {
        FileHandler fh = new FileHandler("simple.log");
        fh.setFormatter(new SimpleFormatter());
        sl.logger.addHandler(fh);
        sl.logger.setLevel(Level.CONFIG);
        sl.logMessages();
    } catch (IOException e){
        e.getMessage();
    }
}
```

ORACLE
Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Once you create a handler, you assign the formatter of your choice as shown in the slide.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.
Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Logger Example: `simple.log`

```
Dec 21, 2011 7:44:49 PM com.example.SimpleLogging logMessages
SEVERE: A very severe problem has occured
Dec 21, 2011 7:44:49 PM com.example.SimpleLogging logMessages
WARNING: Warning, something bad may be happening
Dec 21, 2011 7:44:49 PM com.example.SimpleLogging logMessages
INFO: Here is INFO you may want to know about
Dec 21, 2011 7:44:49 PM com.example.SimpleLogging logMessages
CONFIG: Here is some info about your CONFIG
```

ORACLE
Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

The SimpleFormatter produces simple single-line messages like those shown in the slide. Notice that, because the logging level was set to CONFIG, four messages are included in this log file.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Quiz

Which two are default logging formatters?

a. SimpleFormatter
b. XMLFormatter
c. BasicFormatter
d. HTMLFormatter

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

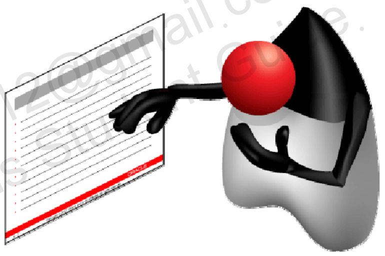
ORACLE

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Answer: a, b

Topics

- Logging API overview
- Logging configuration
- Logging examples
- Configuration files and custom handlers



Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

File-Based Configuration

Logging configuration can be set up using a configuration file.

- The file is read at startup.
- It is a standard `java.util.Properties` file.
- A default sample file can be found in:
  `C:\ProgramFiles\Java\jdk1.7.0_03\jre\lib\logging.properties`

ORACLE
Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Logging configuration can be controlled from a Java Properties file, `logging.properties`. An example file is included in:
`C:\ProgramFiles\Java\jdk1.7.0_03\jre\lib\logging.properties`

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.
Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Custom Log Handler

You can write your own handler to format logging messages.

- Extend the `Handler` class.
- Implement the `publish`, `flush` and `close` methods.
- `CustomLogHandler.java`

Sample output:

```
0 Dec 21, 2011 8:53:33 PM SEVERE A very severe problem has
occured com.example.CustomLogging logMessages
1 Dec 21, 2011 8:53:33 PM WARNING Warning, something bad may be
happening com.example.CustomLogging logMessages
2 Dec 21, 2011 8:53:33 PM INFO Here is INFO you may want to know
about com.example.CustomLogging logMessages
```

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

ORACLE

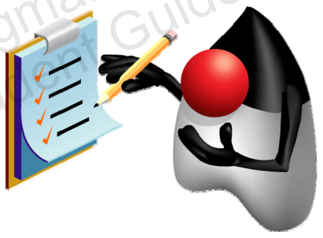
Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

If you want more control over the information included in the log, you can create your own log handler.

Summary

In this lesson, you should have learned how to:

- Add a logging mechanism to your Java application
- Log messages to a file at an appropriate level
- Configure your logger
- Configure logging formatters and handlers



Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.

Practice 16: Overview

Practice 16-1: Logging in Java Applications

Copyright © 2012, Oracle and/or its affiliates. All rights reserved.

Unauthorized reproduction or distribution prohibited. Copyright© 2020, Oracle and/or its affiliates.

Adolfo De+la+Rosa (adolfodelarosa2012@gmail.com) has a non-transferable license to use this Student Guide.