CS601: Principles of Software Development

Logger.

Olga A. Karpenko

Parts of this presentations are based on the materials of Prof. Engle.

Debugging Approaches

- Any large piece of code is likely to have bugs
- How can we debug it?
- Approaches
  - println() statements
  - Debugger
  - Logger

Debugging Approaches

- println() statements
  - Easy to add
  - Hard to remove
  - Poor for multithreading

Debugging Approaches

- A debugger
  - More powerful; allows to examine the state of the system at any point
  - Multithreading bugs may not show up
    - Sensitive to the timing
    - Running the program in debug mode changes the timing

Debugging Approaches

- Using a logger
  - Logging means "recording an activity"
  - Usually efficient
  - Easy to disable
  - Good for multithreading

Issues with Log Statements

- Increase the size of the code
- Reduce its speed
  - Even when logging is turned off

log4j2

- A logging library by Apache
- Efficient
- Configurable
  - without modifying source code

Simple log4j2 Example

```
Logger log = LogManager.getRootLogger();
log.debug("Hello world!");
```

http://logging.apache.org/log4j/2.0/manual/api.html

Logging Levels

- TRACE
  - Used for fine-grained debug messages
  - Usually not shown unless really necessary
- DEBUG
  - Used for normal debug messages
  - Most commonly used level for logging

Logging Levels

- INFO
  - Used for informational messages
  - Often used for major successful events
- WARN
  - Used when something concerning happened

Logging Levels

- ERROR
  - When a possibly recoverable error occurred
  - Nearly always shown to the user

- FATAL
  - Used when an irrecoverable error occurred
  - Often used when about to exit prematurely

Logging Levels

- ALL
  - Turns on all levels of logging, including TRACE
  - Used to turn on all logging for debugging purposes

- OFF
  - Turns off all levels of logging
  - Used to disable logging, speeding up code

Logging Levels

- TRACE « lowest level, rarely used
- DEBUG « most common
- INFO « informational
- WARN « warnings
- ERROR « errors/exceptions
- FATAL « highest level, rarely used

Configuration

- Done via an xml file
- Controls what information is output
- Specifies which levels are output
- Specifies where the messages are output
  - console, file, ...

http://logging.apache.org/log4j/2.0/manual/configuration.html

Log4j Configuration

- Appenders
  - Control where log messages are output
  - Commonly the console and a log file
- Layouts
  - Specify what information is output
  - Commonly the timestamp, level, and message

Log4j Configuration

- Loggers
  - Identified by a name (often class name)
  - Specify level of message to send to an appender
- Root Logger
  - Default logger, always accessible
  - Uses console appender and outputs ERROR messages by default

Sample Configuration

```
<Configuration status="WARN">
        <Appenders>
            <Console name="Console" target="SYSTEM_OUT">
                        <PatternLayout pattern="%m%n">
                        </PatternLayout>
            </Console>
        </Appenders>
        <Loggers>
                <Root level="error">
                        <AppenderRef ref="Console"/>
                </Root>
        </Loggers>

</Configuration>
```

Pattern Layout

- **%level** : level of the logging event

  %level{length=1} to use a single letter
  %level{lowerCase=true} to convert to lowercase
- **%m** : log message
- **%n** : line separator (\n or \r\n)

Pattern Layout

- **%date**{pattern} : timestamp for logging event
  - %d{HH:mm:ss:SSS} to output just time
- **%t** : thread name
- **%file** : name of the file
- **%location** : location where logging event created
  - Expensive operation, use with caution
  - See also %logger{}, %class{}, %method, and %line

Installing log4j2

- Download latest log4j2* jar files:

http://www.apache.org/dyn/closer.lua/logging/log4j/2.6.2/apache-log4j-2.6.2-bin.zip

- Extract and save jar files in some folder
  - Create a "libraries" folder for 3d party libraries
- Create a new Java user library in IntelliJ

"

Installing log4j2

- Click "Add External JARs…"
  - Add log4j-api-2.6.2.jar
  - Add log4j-core-2.6.2.jar
- Attach Javadoc JAR files (recommended)
- Add new user library to build path of project