Logging

JabRef uses the logging facade SLF4j. All log messages are passed internally to tinylog which handles any filtering, formatting and writing of log messages.

Obtaining a logger for a class:

```
private static final Logger LOGGER = LoggerFactory.getLogger(<ClassName>.class);
```

Please always use `LOGGER.debug` for debugging.

Example:

```
String example = "example";
LOGGER.debug("Some state {}", example);
```

Enable logging in `tinylog.properties`:

```
level@org.jabref.example.ExampleClass = debug
```

If the logging event is caused by an exception, please add the exception to the log message as:

```
  catch (SomeException e) {
     LOGGER.warn("Warning text.", e);
     ...
  }
```

When running tests, `tinylog-test.properties` is used. It is located under `src/test/resources`. As default, only `info` is logged. When developing, it makes sense to use `debug` as log level. One can change the log level per class using the pattern `level@class=debug` is set to `debug`. In the `.properties` file, this is done for `org.jabref.model.entry.BibEntry`.

Further reading

SLF4J also support parameterized logging, e.g. if you want to print out multiple arguments in a log statement use a pair of curly braces (`{}`). Head to https://www.slf4j.org/faq.html#logging_performance for examples.