**Code smell** 1- Records should be used instead of ordinary classes when representing immutable data structure

Code snippet:

Text

Description automatically generated

Location of the code: src/main/java/jabref/logic/remote/shared/event/ConnectionLostEvent.java

In this specific piece of code there is the opportunity to introduce records which represent immutable read-only data structure and should be used instead of creating immutable classes. This code smell can easily be fixed by refactoring the class declaration to “record ConnectionLostEvent(BibDatabaseContext bibDatabaseContext)”.

Refactoring proposal:

Graphical user interface, text

Description automatically generated

**Code smell** 2- Records should be used instead of ordinary classes when representing immutable data structure

Code snippet:

Text

Description automatically generated

Location of the code: src/main/java/jabref/logic/remote/shared/event/SharedEntriesNotPresentEvent.java

Again, this piece of code shows the missed opportunity to introduce records which represent immutable read-only data structure and should be used instead of creating immutable classes. This can be fixed by refactoring the class declaration to use “record SharedEntriesNotPresentEvent(List<BibEntry> bibEntries)”.

Refactoring proposal:

Text

Description automatically generated

**Code smell** 3- Records should be used instead of ordinary classes when representing immutable data structure

Code snippet:

Text

Description automatically generated

Location of the code: src/main/java/jabref/logic/shared/event/UpdateRefusedEvent.java

The problem that was described before can also be found in this chunk of code. Once again, the fix is to use “record UpdateRefusedEvent(BibDatabaseContext bibDatabaseContext, BibEntry localBibEntry, BibEntry sharedBibEntry)”, instead of the used class declaration.

Refactoring proposal:

Text

Description automatically generated