Report for Assignment 1

Project chosen

Name: jabref

URL: https://github.com/JabRef/jabref

Number of lines of code and the tool used to count it: 180 kloc, Lizard

Programming language: Java

Coverage measurement

Existing tool

< Inform the name of the existing tool that was executed and how it was executed >
Jacoco.

1. The build.gradle file was configured to include the JaCoCo plugin.

```
plugins {
    id 'jacoco'
}

jacoco {
    toolVersion = "0.8.10"
}

jacocoTestReport {
    dependsOn test
    reports {
        xml.required = true
        csv.required = false
        html.outputLocation = layout.buildDirectory.dir('jacocoHtml')
    }
}
```

2. Run ./gradlew clean test jacocoTestReport

< Show the coverage results provided by the existing tool with a screenshot >

JabRef

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods	Missed	Classes
org.jabref.preferences		3%		0%	796	820	1,979	2,054	602	625	12	21
org.jabref.gui.fieldeditors		6%		4%	563	591	1,346	1,464	306	329	50	56
org.jabref.gui.groups		19%		12%	590	665	1,311	1,619	263	317	14	20
org.jabref.gui.frame		2%		0%	268	277	944	978	138	147	16	18
org.jabref.gui.entryeditor		2%		0%	372	377	1,156	1,184	245	250	27	30
org.jabref.gui.openoffice		0%		0%	357	357	1,157	1,157	204	204	18	18
org.jabref.gui.maintable		9%		5%	356	388	1,028	1,126	215	241	17	24
org.jabref.gui.util		23%		15%	609	751	1,115	1,440	417	534	43	73
org.jabref.gui		8%		4%	399	434	1,077	1,187	260	293	12	21
org.jabref.logic.importer.fetcher		75%		63%	390	1,060	735	2,936	119	588	4	53
org.jabref.logic.shared		9%		4%	230	259	775	855	107	135	6	11
org.jabref.gui.externalfiles		23%		20%	215	284	657	872	128	186	11	23
org.jabref.gui.mergeentries		0%		0%	193	193	591	591	104	104	15	15
org.jabref.logic.importer.fileformat		89%		81%	583	2,263	518	4,967	55	562	2	42
org.jabref.gui.search		0%		0%	131	131	445	445	92	92	13	13
org.jabref.gui.preferences.preview		0%		0%	131	131	406	406	83	83	3	3
org.jabref.gui.importer		2%		2%	140	144	447	462	90	94	15	16
org.jabref.logic.exporter		77%		76%	134	480	455	1,956	46	242	2	27
org.jabref.gui.push		7%		3%	137	146	401	426	91	100	15	17
org.jabref.gui.shared		0%		0%	90	90	336	336	65	65	5	5
org.jabref.logic.bst		73%		67%	221	457	266	1,156	109	252	0	35
org.jabref.cli		46%		31%	141	195	307	612	33	79	1	5
org.jabref.logic.openoffice.backend		0%		0%	121	121	404	404	52	52	6	6
org.jabref.gui.exporter		25%		16%	98	124	333	449	52	75	10	14
org.jabref.gui.linkedfile		22%		15%	106	129	327	428	47	66	6	8
org.jabref.model.openoffice.ootext		2%		4%	101	107	319	333	33	38	3	4
org.jabref.gui.entryeditor.citationrelationtab		18%		20%	74	97	275	344	45	65	2	6
org.jabref.gui.commonfxcontrols		10%		8%	142	155	308	355	104	115	5	10
org.jabref.gui.preferences.network		0%		0%	92	92	278	278	76	76	3	3
org.jabref.logic.importer.fileformat.citavi		15%		20%	375	455	555	644	353	433	1	40
org.jabref.gui.preferences.general		0%		0%	72	72	259	259	51	51	2	2
org.jabref.gui.mergeentries.newmergedialog		30%		19%	107	157	246	391	67	111	6	10
org.jabref.gui.texparser		1%		0%	74	75	244	251	59	60	6	7
org.jabref.model.openoffice.style		11%		12%	138	157	297	344	87	101	10	14
org.jabref.gui.slr		16%		16%	94	108	295	342	69	81	5	7
org.jabref.gui.entryeditor.fileannotationtab		11%		6%	65	75	217	250	41	50	4	5
org.jabref.gui.preferences.table		0%		0%	100	100	227	227	66	66	3	3
org.jabref.logic.undo		0%		n/a	7	7	14	14	7	7	3	3
org.jabref.logic.shared.event		0%		n/a	8	8	16	16	8	8	3	3
org.jabref.model.search		50%		0%	5	11	8	18	3	9	0	1
org.jabref.logic		43%		50%	7	12	13	22	6	11	3	5
org.jabref.model.search.matchers		75%		70%	6	21	8	31	3	16	0	6
org.jabref.logic.remote		86%		93%	3	25	11	66	2	17	0	4
org.jabref.logic.shared.exception		0%		n/a	7	7	14	14	7	7	4	4
org.jabref.model.entry.event		84%		71%	7	25	10	55	3	18	0	5
org.jabref.logic.quality.consistency		96%		73%	7	61	6	171	2	48	0	6
org.jabref.gui.mergeentries.newmergedialog.fieldsmerger		80%		73%	8	27	8	43	2	12	1	5
org.jabref.logic.util.strings		99%		84%	11	66	3	469	1	28	0	10
org.jabref.logic.logging		0%		n/a	6	6	10	10	6	6	1	1
org.jabref.model.openoffice		68%		75%	3	12	3	20	2	10	1	2
org.jabref.logic.texparser		94%		85%	4	36	9	91	0	22	0	2
org.jabref.logic.remote.client		83%		50%	4	9	4	27	0	5	0	1
org.jabref.logic.shared.security		77%		n/a	1	5	2	15	1	5	0	1
org.jabref.gui.logging		0%		n/a	6	6	9	9	6	6	1	1
org.jabref.gui.util.comparator		90%		82%	4	23	5	47	0	6	0	3
org.jabref.logic.auxparser		97%		91%	3	41	3	107	0	24	0	2
org.jabref.model.database.event		82%		n/a	3	11	5	22	3	11	1	5
org.jabref.logic.preview		0%		n/a	1	1	1	1	1	1	1	1
org.jabref.logic.pseudonymization		97%		83%	2	18	3	43	1	15	0	3
org.jabref.model.schema		90%		100%	0	8	3	23	0	5	0	1
org.jabref.logic.push		97%		70%	3	9	1	16	0	4	0	1
org.jabref.model.metadata.event		66%		n/a	1	2	1	4	1	2	0	1
org.jabref.model.groups.event		66%		n/a	1	2	1	4	1	2	0	1
org.jabref.http		0%		n/a	1	1	1	1	1	1	1	1
org.jabref.logic.formatter		100%		100%	0	22	0	34	0	14	0	2
org.jabref.logic.help		100%		n/a	0	3	0	44	0	3	0	1
org.jabref.logic.formatter.minifier		100%		100%	0	20	0	31	0	13	0	2
org.jabref.logic.importer.fileformat.medline		100%		n/a	0	5	0	5	0	5	0	5
org.jabref.logic.importer.fileformat.mods		100%		n/a	0	5	0	6	0	5	0	3
org.jabref.model.paging		100%		n/a	0	6	0	11	0	6	0	1
org.jabref.logic.groups		100%		n/a	0	1	0	3	0	1	0	1
Total	170,530 of 327,443	47%	10,908 of 22,031	50%	15,642	26,144	36,689	66,723	8,757	14,663	781	1,784

Your own coverage tool

<The following is supposed to be repeated for each group member>

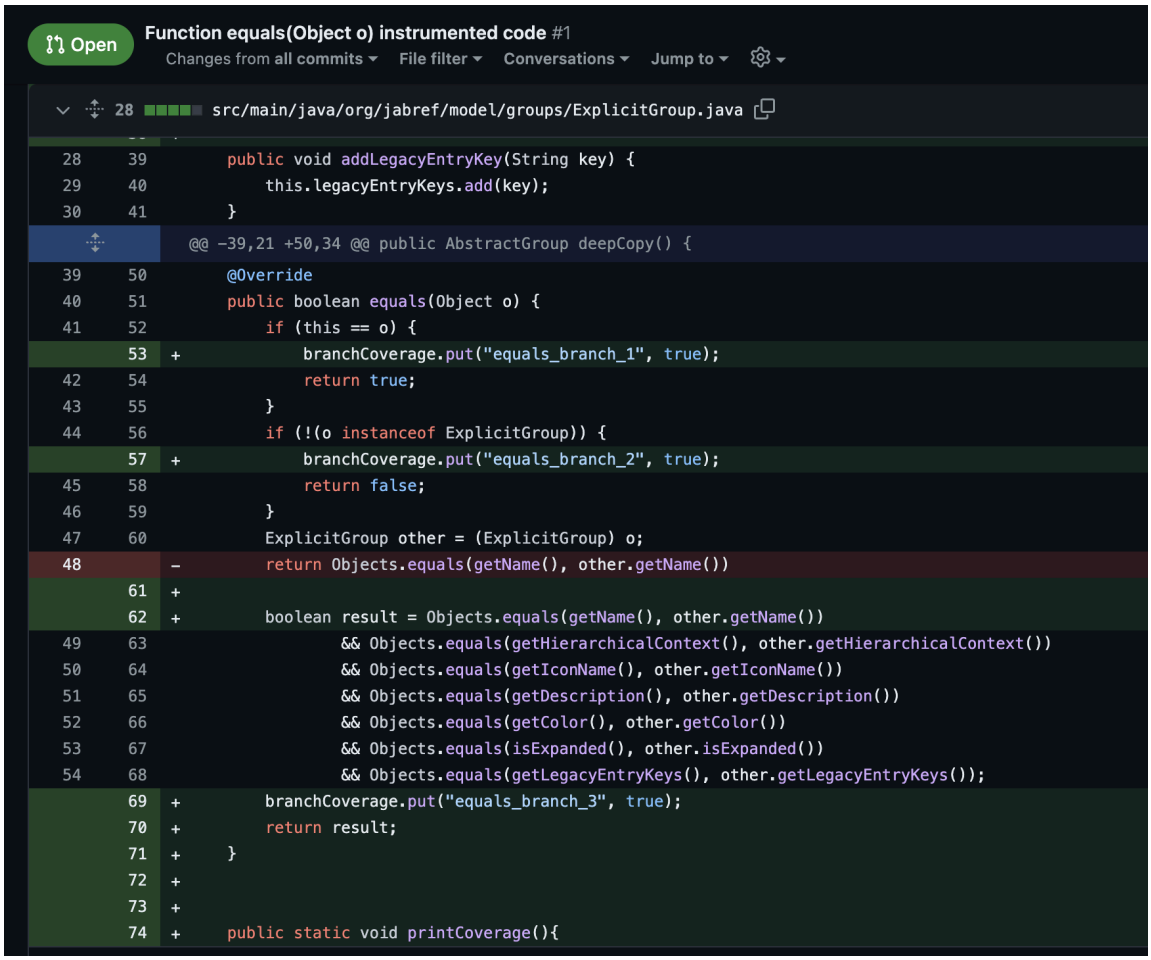
< Xingyun Wang >

< Function 1 name >
equals()

Path : jabref > src > main > java > org > jabref > model > groups > ExplicitGroup > equals

< Show a patch (diff) or a link to a commit made in your forked repository that shows the instrumented code to gather coverage measurements >

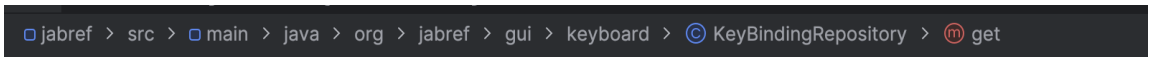
https://github.com/xingyun-w/jabref/pull/1/commits/6d9946f3233a492e86331d573c4c2f70018186e6



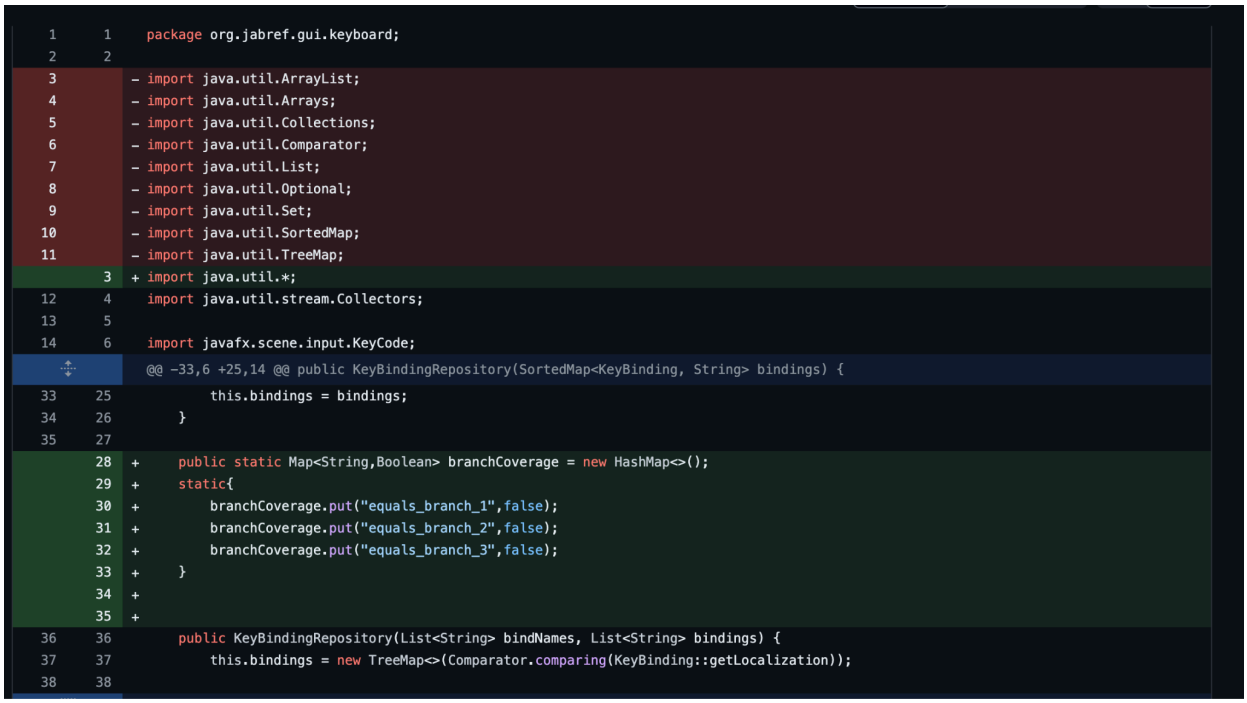
< Provide a screenshot of the coverage results output by the instrumentation >

/Run ./gradlew test --tests "org.jabref.model.groups.ExplicitGroupTest"
to see print messages.

< Function 2 name >
get()

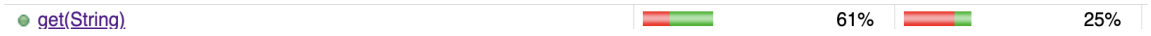


< Show a patch (diff) or a link to a commit made in your forked repository that shows the instrumented code to gather coverage measurements >
https://github.com/JabRef/jabref/commit/0a7060ee314f2edd5230ad970bb4c40804326007



```
1   1   package org.jabref.gui.keyboard;
2   2
3       - import java.util.ArrayList;
4       - import java.util.Arrays;
5       - import java.util.Collections;
6       - import java.util.Comparator;
7       - import java.util.List;
8       - import java.util.Optional;
9       - import java.util.Set;
10      - import java.util.SortedMap;
11      - import java.util.TreeMap;
        3 + import java.util.*;
12  4   import java.util.stream.Collectors;
13  5
14  6   import javafx.scene.input.KeyCode;
    ↕   @@ -33,6 +25,14 @@ public KeyBindingRepository(SortedMap<KeyBinding, String> bindings) {
33  25          this.bindings = bindings;
34  26      }
35  27
        28  +   public static Map<String,Boolean> branchCoverage = new HashMap<>();
        29  +   static{
        30  +       branchCoverage.put("equals_branch_1",false);
        31  +       branchCoverage.put("equals_branch_2",false);
        32  +       branchCoverage.put("equals_branch_3",false);
        33  +   }
        34  +
        35  +
36  36      public KeyBindingRepository(List<String> bindNames, List<String> bindings) {
37  37          this.bindings = new TreeMap<>(Comparator.comparing(KeyBinding::getLocalization));
38  38
```

< Provide a screenshot of the coverage results output by the instrumentation >
./gradlew test --tests "org.jabref.gui.keyboard.KeyBindingsTabModelTest"



< Yiyang Sun >

< Function 1 name >

getMainFileDirectory()

Path:
JabRef > org.jabref.preferences > FilePreferences

< Show a patch (diff) or a link to a commit made in your forked repository that shows the instrumented code to gather coverage measurements >
https://github.com/xingyun-w/jabref/commit/87927dd030eef4bee7ac00cabca5843420eb9fbc

```
39  +     private static final Map<String,Boolean> branchCov = new HashMap<>();
40  +
41  +     static {
42  +         branchCov.put("getMainFD_0", false);
43  +         branchCov.put("getMainFD_1", false);
44  +     }
45  +
46        public FilePreferences(String userAndHost,
47                               String mainFileDirectory,
48                               boolean storeFilesRelativeToBibFile,
     @@ -77,12 +81,25 @@ public StringProperty getUserAndHostProperty() {
81
82        public Optional<Path> getMainFileDirectory() {
83            if (StringUtil.isBlank(mainFileDirectory.getValue())) {
84  +             branchCov.put("getMainFD_0", true);
85                return Optional.empty();
86            } else {
87  +             branchCov.put("getMainFD_1", true);
88                return Optional.of(Path.of(mainFileDirectory.getValue()));
89            }
90        }
91
92  +     public static void printCov() {
93  +         for (Map.Entry <String, Boolean> entry : branchCov.entrySet()){
94  +             System.out.println((entry.getKey() + " was " + (entry.getValue()
95  +         }
96  +     }
97  +
98  +     public static void  resetBranchCov() {
99  +         branchCov.put("getMainFD_0", false);
100 +         branchCov.put("getMainFD_1", false);
101 +     }
102 +
```

src/test/java/org/jabref/preferences/FilePreferencesTest.java

```
...   @@ -0,0 +1,24 @@
1   + package org.jabref.preferences;
2   + import org.junit.jupiter.api.Test;
3   +
4   + import java.util.Set;
5   +
6   + public class FilePreferencesTest {
7   +
8   +     @Test
9   +     public void testGetMainFD(){
10  +         FilePreferences filePreferences = new FilePreferences("user", "", false, "", "", false, false,
      null, Set.of(), false, null, false, false);
11  +
12  +         filePreferences.setMainFileDirectory("");
13  +         filePreferences.getMainFileDirectory();
14  +         System.out.println("testing getMainFD 0 branch:");
15  +         FilePreferences.printCov();
16  +
17  +         FilePreferences.resetBranchCov();
18  +
19  +         filePreferences.setMainFileDirectory("/path/path2");
20  +         filePreferences.getMainFileDirectory();
21  +         System.out.println("testing getMainFD 1 branch:");
22  +         FilePreferences.printCov();
23  +     }
24  + }
```

< Provide a screenshot of the coverage results output by the instrumentation >

Element	Missed Instructions	Cov.	Missed Branches	Cov.
getMainFileDirectory()		0%		0%

Tests | Standard output

```
testing getMainFD 0 branch:
getMainFD_0 was hit
getMainFD_1 was not hit
testing getMainFD 1 branch:
getMainFD_0 was not hit
getMainFD_1 was hit
```

Run:
./gradlew clean test --tests org.jabref.preferences.FilePreferencesTest

Check printout info:
build/reports/test/index.html

< Function 2 name >
getWriteMetadatatoPdf()

Path:
JabRef > org.jabref.cli > JabRefCLI

< Show a patch (diff) or a link to a commit made in your forked repository that shows the instrumented code to gather coverage measurements >
https://github.com/JabRef/jabref/commit/b9473f6448abe68e9aa53c67b50c597f6ad45175

```
34 +    private static final Map<String, Boolean> BRANCH_COV = new HashMap<>();
35 +
36 +    static {
37 +        BRANCH_COV.put("brand_00", false);
38 +        BRANCH_COV.put("brand_01", false);
39 +        BRANCH_COV.put("brand_02", false);
40 +        BRANCH_COV.put("brand_03", false);
41 +    }
42 +
```

```
177         public String getWriteMetadatatoPdf() {
178 +           if ( cl.hasOption("writeMetadatatoPdf") ) {
179 +               BRANCH_COV.put("brand_00", true);
180 +           } else if ( cl.hasOption("writeXMPtoPdf") ) {
181 +               BRANCH_COV.put("brand_01", true);
182 +           } else if ( cl.hasOption("embeddBibfileInPdf") ) {
183 +               BRANCH_COV.put("brand_02", true);
184 +           } else {
185 +               BRANCH_COV.put("brand_03", true);
186 +           }
187 +
```

```
193 +     public static void printCov() {
194 +         BRANCH_COV.forEach((branch, hit) -> System.out.println( branch + " was " + ( hit ? "hit" : "not
    hit" )));
195 +     }
196 +
197 +     public static void resetCov() {
198 +         BRANCH_COV.replaceAll( (branch, hit) -> false );
199 +     }
200 +
```

< Provide a screenshot of the coverage results output by the instrumentation >

Element	Missed Instructions	Cov.	Missed Branches	Cov.
getWriteMetadatatoPdf()		0%		0%

Tests | Standard output

```
testing branch_03:
brand_00 was not hit
brand_01 was not hit
brand_02 was not hit
brand_03 was hit
testing branch_02:
brand_00 was not hit
brand_01 was not hit
brand_02 was hit
brand_03 was not hit
testing branch_00:
brand_00 was hit
brand_01 was not hit
brand_02 was not hit
brand_03 was not hit
testing branch_01:
brand_00 was not hit
brand_01 was hit
brand_02 was not hit
brand_03 was not hit
```

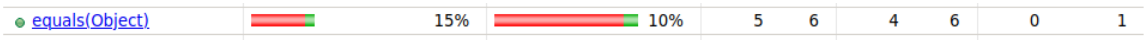
Run:
./gradlew clean test --tests org.jabref.cli.JabRefCLITest

Check printout info:
build/reports/test/index.html

< YuliWang >

< Function 1 name >

public boolean equals(Object o)
src/main/java/org/jabref/gui/theme/Theme-public boolean equals(Object o)



< Show a patch (diff) or a link to a commit made in your forked repository that shows the instrumented code to gather coverage measurements >

src/main/java/org/jabref/gui/theme/Theme.java

```
@@ -2,6 +2,8 @@

 import java.util.Objects;
 import java.util.Optional;
+import java.util.HashMap;
+import java.util.Map;

 /**
  * Represents one of three types of a css based Theme for JabRef:
@@ -22,6 +24,13 @@ public enum Type {
     private final String name;
     private final Optional<StyleSheet> additionalStylesheet;

+    public static Map<String, Boolean> branchCoverage = new HashMap<>();
+    static {
+        branchCoverage.put("this == o", false);
+        branchCoverage.put("o == null", false);
+        branchCoverage.put("getClass() != o.getClass()", false);
+    }
+
     public Theme(String name) {
         Objects.requireNonNull(name);

@@ -95,9 +104,15 @@ public Optional<StyleSheet> getAdditionalStylesheet() {
     @Override
     public boolean equals(Object o) {
         if (this == o) {
+            branchCoverage.put("this == o", true);
             return true;
         }
-        if (o == null || getClass() != o.getClass()) {
+        if (o == null) {
+            branchCoverage.put("o == null", true);
+            return false;
+        }
+        if (getClass() != o.getClass()) {
+            branchCoverage.put("getClass() != o.getClass()", true);
             return false;
         }
         Theme that = (Theme) o;
@@ -116,4 +131,11 @@ public String toString() {
                 ", name='" + name + '\'' +
                 '}';
     }
+
+    public static void printCoverage() {
+        for (Map.Entry<String, Boolean> entry : branchCoverage.entrySet()) {
+            System.out.println("Branch \"" + entry.getKey() + "\" was " + (entry.getValue() ? "hit" : "not hit"));
+        }
+    }
+
 }
```

src/test/java/org/jabref/gui/theme/ThemeTest.java

```
@@ -2,12 +2,19 @@

 import java.util.Optional;

+import org.junit.jupiter.api.AfterAll;
 import org.junit.jupiter.api.Test;

 import static org.junit.jupiter.api.Assertions.assertEquals;
 import static org.junit.jupiter.api.Assertions.assertTrue;

 public class ThemeTest {
+
+    @AfterAll
+    public static void printCoverage() {
+        Theme.printCoverage();
+    }
+
     @Test
     public void lightThemeUsedWhenPathIsBlank() {
         Theme theme = new Theme("");
```

< Provide a screenshot of the coverage results output by the instrumentation >
./gradlew test --tests "org.jabref.gui.theme.ThemeTest"

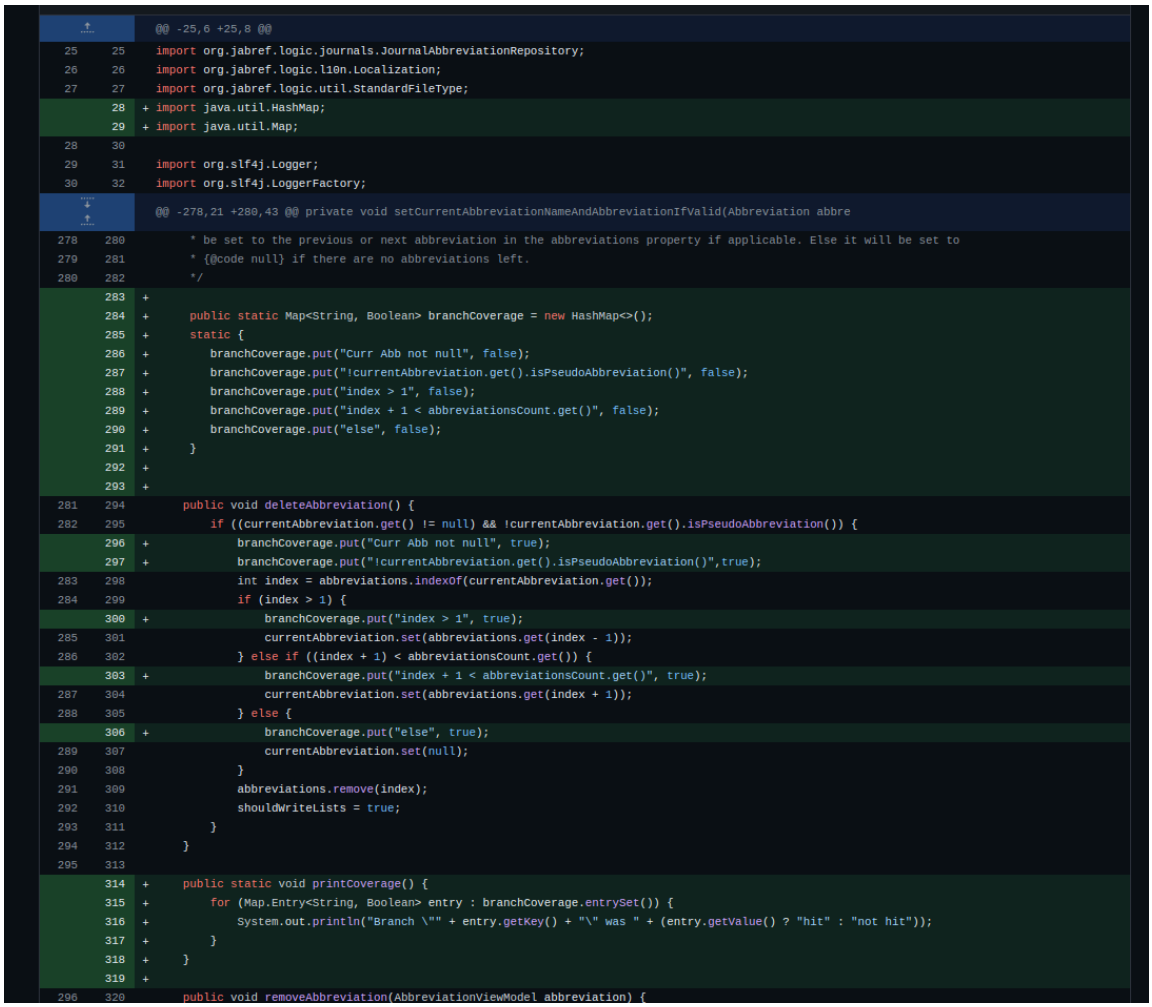
```
ThemeTest STANDARD_OUT
    Branch "this == o" was not hit
    Branch "o == null" was not hit
    Branch "getClass() != o.getClass()" was not hit
```

< Function 2 name >

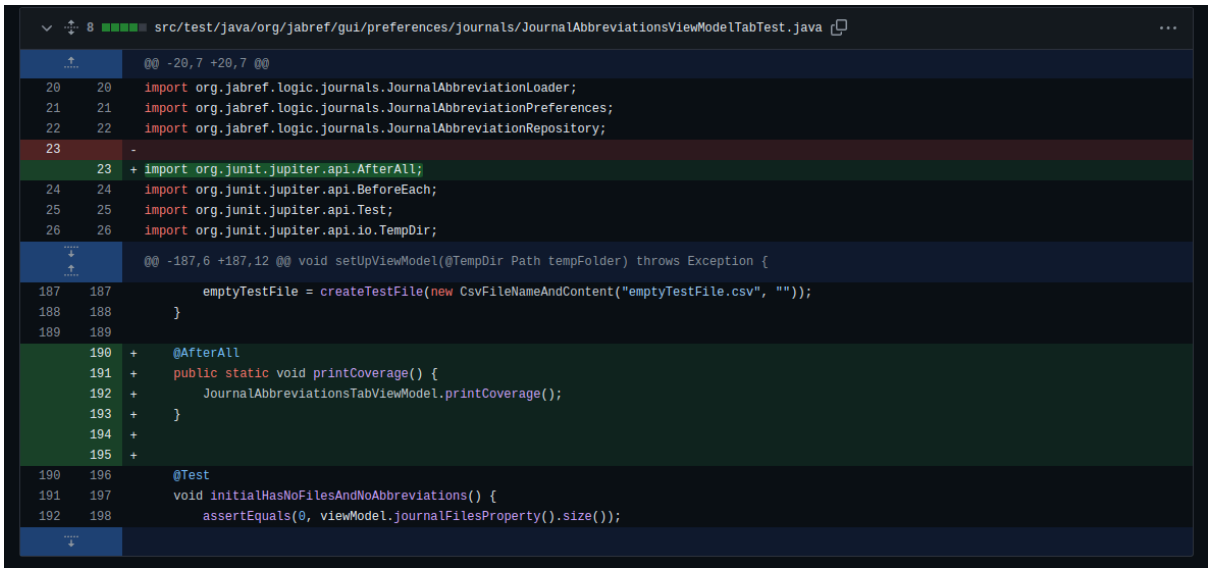
void deleteAbbreviation()

src/main/java/org/jabref/gui/preferences/journals/JournalAbbreviationsTabViewModel.java-public void deleteAbbreviation()

< Show a patch (diff) or a link to a commit made in your forked repository that shows the instrumented code to gather coverage measurements >

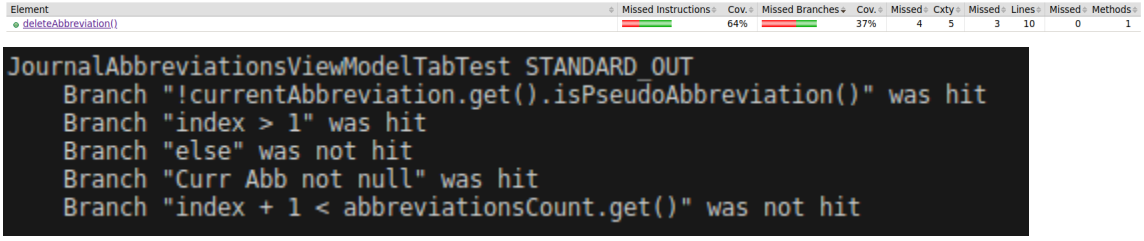


```
@@ -25,6 +25,8 @@
25 25  import org.jabref.logic.journals.JournalAbbreviationRepository;
26 26  import org.jabref.logic.l10n.Localization;
27 27  import org.jabref.logic.util.StandardFileType;
   28 + import java.util.HashMap;
   29 + import java.util.Map;
28 30
29 31  import org.slf4j.Logger;
30 32  import org.slf4j.LoggerFactory;
@@ -278,21 +280,43 @@ private void setCurrentAbbreviationNameAndAbbreviationIfValid(Abbreviation abbre
278 280       * be set to the previous or next abbreviation in the abbreviations property if applicable. Else it will be set to
279 281       * {@code null} if there are no abbreviations left.
280 282       */
   283 +
   284 +     public static Map<String, Boolean> branchCoverage = new HashMap<>();
   285 +     static {
   286 +         branchCoverage.put("Curr Abb not null", false);
   287 +         branchCoverage.put("!currentAbbreviation.get().isPseudoAbbreviation()", false);
   288 +         branchCoverage.put("index > 1", false);
   289 +         branchCoverage.put("index + 1 < abbreviationsCount.get()", false);
   290 +         branchCoverage.put("else", false);
   291 +     }
   292 +
   293 +
281 294      public void deleteAbbreviation() {
282 295          if ((currentAbbreviation.get() != null) && !currentAbbreviation.get().isPseudoAbbreviation()) {
   296 +             branchCoverage.put("Curr Abb not null", true);
   297 +             branchCoverage.put("!currentAbbreviation.get().isPseudoAbbreviation()",true);
283 298              int index = abbreviations.indexOf(currentAbbreviation.get());
284 299              if (index > 1) {
   300 +                 branchCoverage.put("index > 1", true);
285 301                  currentAbbreviation.set(abbreviations.get(index - 1));
286 302              } else if ((index + 1) < abbreviationsCount.get()) {
   303 +                 branchCoverage.put("index + 1 < abbreviationsCount.get()", true);
287 304                  currentAbbreviation.set(abbreviations.get(index + 1));
288 305              } else {
   306 +                 branchCoverage.put("else", true);
289 307                  currentAbbreviation.set(null);
290 308              }
291 309              abbreviations.remove(index);
292 310              shouldWriteLists = true;
293 311          }
294 312      }
295 313
   314 +     public static void printCoverage() {
   315 +         for (Map.Entry<String, Boolean> entry : branchCoverage.entrySet()) {
   316 +             System.out.println("Branch \"" + entry.getKey() + "\" was " + (entry.getValue() ? "hit" : "not hit"));
   317 +         }
   318 +     }
   319 +
296 320      public void removeAbbreviation(AbbreviationViewModel abbreviation) {
```



< Provide a screenshot of the coverage results output by the instrumentation >

./gradlew test --tests "org.jabref.gui.preferences.journals.JournalAbbreviationsViewModelTabTest"



Coverage improvement

Individual tests

<The following is supposed to be repeated for each group member>

< Xingyun Wang >

< Test 1 >

< Show a patch (diff) or a link to a commit made in your forked repository that shows the new/enhanced test >

https://github.com/JabRef/jabref/commit/0a7060ee314f2edd5230ad970bb4c40804326007

See changes in src/test/java/org/jabref/model/groups/ExplicitGroupTest.java

98 src/test/java/org/jabref/model/groups/ExplicitGroupTest.java

```
@@ -16,13 +16,14 @@ class ExplicitGroupTest {
 
     private ExplicitGroup group;
     private ExplicitGroup group2;
-
+    private ExplicitGroup group3;
     private BibEntry entry;
 
     @BeforeEach
     void setUp() {
         group = new ExplicitGroup("myExplicitGroup", GroupHierarchyType.INDEPENDENT, ',');
-        group2 = new ExplicitGroup("myExplicitGroup2", GroupHierarchyType.INCLUDING, ',');
+        group2 = new ExplicitGroup("myExplicitGroup", GroupHierarchyType.INDEPENDENT, ',');  // Same as group
+        group3 = new ExplicitGroup("anotherGroup", GroupHierarchyType.INDEPENDENT, ',');  // Different from group
         entry = new BibEntry();
     }
 
@@ -40,10 +41,10 @@ void addSingleGroupToNonemptyBibEntryAppendsToGroupsField() {
     }
 
     @Test
-    void addTwoGroupsToBibEntryChangesGroupsField() {
-        group.add(entry);
-        group2.add(entry);
-        assertEquals(Optional.of("myExplicitGroup, myExplicitGroup2"), entry.getField(StandardField.GROUPS));
+    void addTwoGroupsToBibEntryChangesGroupsField() {
+     group.add(entry);
+     group2.add(entry);
+     assertEquals(Optional.of("myExplicitGroup, myExplicitGroup2"), entry.getField(StandardField.GROUPS));
     }
 
 }
```

< Provide a screenshot of the old coverage results (the same as you already showed above) >

equals(Object)	88%	55%	8	10	2	16	0	1

< Provide a screenshot of the new coverage results >

equals(Object)	100%	100%	0	10	0	16	0	1

< State the coverage improvement with a number and elaborate on why the coverage is improved >

Improve 45%

```java
    public boolean equals(Object o) {
        if (this == o) {
            branchCoverage.put("equals_branch_1", true);
            return true;
        }
        if (!(o instanceof ExplicitGroup)) {
            branchCoverage.put("equals_branch_2", true);
            return false;
        }
        ExplicitGroup other = (ExplicitGroup) o;

        boolean result = Objects.equals(getName(), other.getName())
                && Objects.equals(getHierarchicalContext(), other.getHierarchicalContext())
                && Objects.equals(getIconName(), other.getIconName())
                && Objects.equals(getDescription(), other.getDescription())
                && Objects.equals(getColor(), other.getColor())
                && Objects.equals(isExpanded(), other.isExpanded())
                && Objects.equals(getLegacyEntryKeys(), other.getLegacyEntryKeys());
        branchCoverage.put("equals_branch_3", true);
        return result;
    }
```

Originally there were no explicit test cases focused on the equals() method. The first few tests use self-comparison, comparisons of different groups, and edge testing to ensure that the first two branches are covered. Then I created test cases targeting the third branch, which covers a variety of attributes, such as name, hierarchical context, and icon, to ensure all specific branches within the equals() method are exercised.

< Test 2 >

< Show a patch (diff) or a link to a commit made in your forked repository that shows the new/enhanced test >

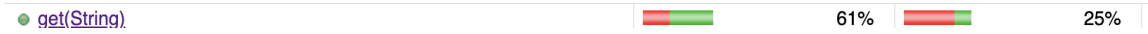
https://github.com/JabRef/jabref/commit/e01d36b7c093e79f92eb839e238c31636eba3b9c

See changes in src/test/java/org/jabref/gui/keyboard/KeyBindingsTabModelTest.java

```
src/test/java/org/jabref/model/groups/ExplicitGroupTest.java

@@ -16,13 +16,14 @@ class ExplicitGroupTest {
 
     private ExplicitGroup group;
     private ExplicitGroup group2;
-
+    private ExplicitGroup group3;
     private BibEntry entry;
 
     @BeforeEach
     void setUp() {
         group = new ExplicitGroup("myExplicitGroup", GroupHierarchyType.INDEPENDENT, ',');
-        group2 = new ExplicitGroup("myExplicitGroup2", GroupHierarchyType.INCLUDING, ',');
+        group2 = new ExplicitGroup("myExplicitGroup", GroupHierarchyType.INDEPENDENT, ',');  // Same as group
+        group3 = new ExplicitGroup("anotherGroup", GroupHierarchyType.INDEPENDENT, ',');  // Different from group
         entry = new BibEntry();
     }
 
@@ -40,10 +41,10 @@ void addSingleGroupToNonemptyBibEntryAppendsToGroupsField() {
     }
 
     @Test
-    void addTwoGroupsToBibEntryChangesGroupsField() {
-        group.add(entry);
-        group2.add(entry);
-        assertEquals(Optional.of("myExplicitGroup, myExplicitGroup2"), entry.getField(StandardField.GROUPS));
+    void addTwoGroupsToBibEntryChangesGroupsField() {
+     group.add(entry);
+     group2.add(entry);
+     assertEquals(Optional.of("myExplicitGroup, myExplicitGroup2"), entry.getField(StandardField.GROUPS));
     }
 
```

< Provide a screenshot of the old coverage results (the same as you already showed above) >



< Provide a screenshot of the new coverage results >

< State the coverage improvement with a number and elaborate on why the coverage is improved >

Improve 75%

```java
    public String get(String key) {
        Optional<KeyBinding> keyBinding = getKeyBinding(key);
        Optional<String> result = keyBinding.flatMap(k -> Optional.ofNullable(bindings.get(k)));

        if (result.isPresent()) {
            branchCoverage.put("equals_branch_1", true);
            return result.get();
        } else if (keyBinding.isPresent()) {
            branchCoverage.put("equals_branch_2", true);
            return keyBinding.get().getDefaultKeyBinding();
        } else {
            branchCoverage.put("equals_branch_3", true);
            return "Not associated";
        }
    }
```

The console messages indicate that only the first branch is being hit. To cover all branches, I have added two test cases: one for handling the case where the result is null, and another for when there is no key binding.

< Yiyang Sun >

<Test 1>

< Show a patch (diff) or a link to a commit made in your forked repository that shows the new/enhanced test >

https://github.com/xingyun-w/jabref/commit/87927dd030eef4bee7ac00cabca5843420eb9fbc

src/test/java/org/jabref/preferences/FilePreferencesTest.java

```
@@ -0,0 +1,24 @@
+ package org.jabref.preferences;
+ import org.junit.jupiter.api.Test;
+
+ import java.util.Set;
+
+ public class FilePreferencesTest {
+
+     @Test
+     public void testGetMainFD(){
+         FilePreferences filePreferences = new FilePreferences("user", "", false, "", "", false, false, null, Set.of(), false, null, false, false);
+
+         filePreferences.setMainFileDirectory("");
+         filePreferences.getMainFileDirectory();
+         System.out.println("testing getMainFD 0 branch:");
+         FilePreferences.printCov();
+
+         FilePreferences.resetBranchCov();
+
+         filePreferences.setMainFileDirectory("/path/path2");
+         filePreferences.getMainFileDirectory();
+         System.out.println("testing getMainFD 1 branch:");
+         FilePreferences.printCov();
+     }
+ }
```

< Provide a screenshot of the old coverage results (the same as you already showed above) >

< Provide a screenshot of the new coverage results >

< State the coverage improvement with a number and elaborate on why the coverage is improved >

The coverage has improved by 100%.

The getMainFileDirectory() function has two possible outcomes: it either returns the correct path or it returns null. I can achieve full coverage by writing tests that cover both scenarios.

< Test 2 >

< Show a patch (diff) or a link to a commit made in your forked repository that shows the new/enhanced test >

https://github.com/JabRef/jabref/commit/b9473f6448abe68e9aa53c67b50c597f6ad45175

```
158 +     private CommandLine cl;
159 +     private JabRefCLI jabRefCLI;
160 +
161 +     @BeforeEach
162 +     void setup() throws ParseException, NoSuchFieldException, IllegalAccessException {
163 +         cl = Mockito.mock(CommandLine.class);
164 +         jabRefCLI = new JabRefCLI(new String[0]);
165 +
166 +         java.lang.reflect.Field clField = JabRefCLI.class.getDeclaredField("cl");
167 +         clField.setAccessible(true);
168 +
169 +         clField.set(jabRefCLI, cl);
170 +     }
171 +
172 +     @Test
173 +     void testGetWriteMetadatatoPdf_WriteMetadatatoPdf() {
174 +         when(cl.hasOption("writeMetadatatoPdf")).thenReturn(true);
175 +         when(cl.getOptionValue("writeMetadatatoPdf")).thenReturn("Metadata_value");
176 +
177 +         assertEquals("Metadata_value", jabRefCLI.getWriteMetadatatoPdf());
178 +
179 +         System.out.println("testing branch_00:");
180 +         JabRefCLI.printCov();
181 +         JabRefCLI.resetCov();
182 +     }
```

```
184 +     @Test
185 +     void testGetWriteMetadatatoPdf_WriteXMPtoPdf() {
186 +         when(cl.hasOption("writeMetadatatoPdf")).thenReturn(false);
187 +         when(cl.hasOption("writeXMPtoPdf")).thenReturn(true);
188 +         when(cl.getOptionValue("writeXMPtoPdf")).thenReturn("XMP_value");
189 +
190 +         assertEquals("XMP_value", jabRefCLI.getWriteMetadatatoPdf());
191 +
192 +         System.out.println( "testing branch_01: " );
193 +         JabRefCLI.printCov();
194 +         JabRefCLI.resetCov();
195 +     }
196 +
197 +     @Test
198 +     void testGetWriteMetadatatoPdf_EmbeddBibfileInPdf() {
199 +         when(cl.hasOption("writeMetadatatoPdf")).thenReturn(false);
200 +         when(cl.hasOption("writeXMPtoPdf")).thenReturn(false);
201 +         when(cl.hasOption("embeddBibfileInPdf")).thenReturn(true);
202 +         when(cl.getOptionValue("embeddBibfileInPdf")).thenReturn("embeddBibfile_value");
203 +
204 +         assertEquals("embeddBibfile_value", jabRefCLI.getWriteMetadatatoPdf());
205 +
206 +         System.out.println( "testing branch_02: " );
207 +         JabRefCLI.printCov();
208 +         JabRefCLI.resetCov();
209 +     }
210 +
211 +     @Test
212 +     void testGetWriteMetadatatoPdf_Null() {
213 +         when(cl.hasOption("writeMetadatatoPdf")).thenReturn(false);
214 +         when(cl.hasOption("writeXMPtoPdf")).thenReturn(false);
215 +         when(cl.hasOption("embeddBibfileInPdf")).thenReturn(false);
216 +
217 +         assertNull(jabRefCLI.getWriteMetadatatoPdf());
218 +
219 +         System.out.println( "testing branch_03: " );
220 +         JabRefCLI.printCov();
221 +         JabRefCLI.resetCov();
222 +     }
```

< Provide a screenshot of the old coverage results (the same as you already showed above) >

Element	Missed Instructions	Cov.	Missed Branches	Cov.
getWriteMetadatatoPdf()		0%		0%

< Provide a screenshot of the new coverage results >

Element	Missed Instructions	Cov.	Missed Branches	Cov.
getWriteMetadatatoPdf()		100%		100%

< State the coverage improvement with a number and elaborate on why the coverage is improved >

The coverage has improved by 100%.

There are four possible data scenarios for getWriteMetadatatoPdf(): Metadata, XMP, embeddBibfile, and null. Therefore, four tests need to be written to cover these scenarios. Additionally, before each test, it is essential to create a JabRefCLI object and mock the command line because getWriteMetadatatoPdf() retrieves its information from the command line.

< Yuli Wang >

<Test 1>

for the Theme.euqals

< Show a patch (diff) or a link to a commit made in your forked repository that shows the new/enhanced test >

```
63  71          assertEquals("idonotexist.css", theme.getAdditionalStylesheet().get().getWatchPath().getFileName().toString());
64  72      }
    73  +
    74  +   @Test
    75  +   public void thisEqualsO() {
    76  +       Theme theme = new Theme("test");
    77  +       assertTrue(theme.equals(theme));
    78  +   }
    79  +
    80  +   @Test
    81  +   public void oEqualsNull() {
    82  +       Theme theme = new Theme("test");
    83  +       assertFalse(theme.equals(null));
    84  +   }
    85  +
    86  +   @Test
    87  +   public void differentClass() {
    88  +       Theme theme = new Theme("test");
    89  +       String differentClassObject = "test";
    90  +       assertFalse(theme.equals(differentClassObject));
    91  +   }
    92  +
    93  +   @Test
    94  +   public void differentType() {
    95  +       Theme theme1 = new Theme("test1");
    96  +       Theme theme2 = new Theme("test2");
    97  +       assertFalse(theme1.equals(theme2));
    98  +   }
    99  +
   100  +   @Test
   101  +   public void sameTypeAndName() {
   102  +       Theme theme1 = new Theme("test");
   103  +       Theme theme2 = new Theme("test");
   104  +       assertTrue(theme1.equals(theme2));
   105  +   }
   106  +
   107  +   @Test
   108  +   public void sameAttributesDifferentObjects() {
   109  +       Theme theme1 = new Theme("test");
   110  +       Theme theme2 = new Theme("test");
   111  +       assertTrue(theme1.equals(theme2));
   112  +   }
   113  +
   114  +   @Test
   115  +   public void sameTypeDifferentName() {
   116  +       Theme theme1 = new Theme("test1");
   117  +       Theme theme2 = new Theme("test2");
   118  +       assertFalse(theme1.equals(theme2));
   119  +   }
   120  +
65  121 }
```

< Provide a screenshot of the old coverage results (the same as you already showed above) >

equals(Object)	15%	10%	5	6	4	6	0	1

< Provide a screenshot of the new coverage results >

equals(Object)	100%	90%	1	6	0	11	0	1

< State the coverage improvement with a number and elaborate on why the coverage is improved >

The coverage improved by 80 percent.

For the equals method, there's 4 main conditional branches: "this == o","o == null","getClass() != 0.getClass()" and "type & name things", thus, there's at most 7 specific case for these 4 branches: self-comparison for "this == o",compare with null for "o == null",compare with object and different class for "getClass() != 0.getClass()", and the other 4 kinds of same types & names, same types different names, same names different types, and different sames & types; by applying these test functions, the coverage improves from 10% to 90%.

<Test 2>

< Show a patch (diff) or a link to a commit made in your forked repository that shows the new/enhanced test >

```
// CurrAbb is null
@Test
void testDelAbbCurIsNull() {
    viewModel.currentAbbreviationProperty().set(null);
    viewModel.deleteAbbreviation();
    assertNull(viewModel.currentAbbreviationProperty().get());
}

// CurrAbb is pseudo
@Test
void testDelAbbCurIsPseudo() {
    AbbreviationViewModel pseudoAbbreviation = mock(AbbreviationViewModel.class);
    when(pseudoAbbreviation.isPseudoAbbreviation()).thenReturn(true);

    viewModel.currentAbbreviationProperty().set(pseudoAbbreviation);
    viewModel.deleteAbbreviation();
    assertEquals(pseudoAbbreviation, viewModel.currentAbbreviationProperty().get());
}

// CurrAbb is not Null nor Pseudo, index>1
@Test
void testDelAbbCurIsNotNulPseudoIndexMore() {
    AbbreviationViewModel abbreviation1 = new AbbreviationViewModel(new Abbreviation("Test1", "T1"));
    AbbreviationViewModel abbreviation2 = new AbbreviationViewModel(new Abbreviation("Test2", "T2"));
    AbbreviationViewModel abbreviation3 = new AbbreviationViewModel(new Abbreviation("Test3", "T3"));

    viewModel.abbreviationsProperty().addAll(abbreviation1, abbreviation2, abbreviation3);
    viewModel.currentAbbreviationProperty().set(abbreviation2);
    viewModel.deleteAbbreviation();

    assertEquals(abbreviation1, viewModel.currentAbbreviationProperty().get());
    assertFalse(viewModel.abbreviationsProperty().contains(abbreviation2));
}
```

```java
// CurrAbb is not Null nor Pseudo, index<=1
@Test
void testDelAbbCurIsNotNulPseudoIndexLess() {
    AbbreviationViewModel abbreviation1 = new AbbreviationViewModel(new Abbreviation("Test1", "T1"));
    AbbreviationViewModel abbreviation2 = new AbbreviationViewModel(new Abbreviation("Test2", "T2"));

    viewModel.abbreviationsProperty().addAll(abbreviation1, abbreviation2);
    viewModel.currentAbbreviationProperty().set(abbreviation1);
    viewModel.deleteAbbreviation();

    assertEquals(abbreviation2, viewModel.currentAbbreviationProperty().get());
    assertFalse(viewModel.abbreviationsProperty().contains(abbreviation1));
}

// CurrAbb is not Null nor Pseudo, index + 1 < abbreviationsCount.get()
@Test
void testDelAbbCurIsNotNulPseudoIndexLessAbbCount() {
    AbbreviationViewModel abbreviation1 = new AbbreviationViewModel(new Abbreviation("Test1", "T1"));
    AbbreviationViewModel abbreviation2 = new AbbreviationViewModel(new Abbreviation("Test2", "T2"));
    AbbreviationViewModel abbreviation3 = new AbbreviationViewModel(new Abbreviation("Test3", "T3"));

    viewModel.abbreviationsProperty().addAll(abbreviation1, abbreviation2, abbreviation3);
    viewModel.currentAbbreviationProperty().set(abbreviation1);
    viewModel.deleteAbbreviation();

    assertEquals(abbreviation2, viewModel.currentAbbreviationProperty().get());
    assertFalse(viewModel.abbreviationsProperty().contains(abbreviation1));
}

// CurrAbb is not Null nor Pseudo, index + 1 >= abbreviationsCount.get()
@Test
void estDelAbbCurIsNotNulPseudoIndexMoreAbbCount() {
    AbbreviationViewModel abbreviation1 = new AbbreviationViewModel(new Abbreviation("Test1", "T1"));
    AbbreviationViewModel abbreviation2 = new AbbreviationViewModel(new Abbreviation("Test2", "T2"));

    viewModel.abbreviationsProperty().addAll(abbreviation1, abbreviation2);
    viewModel.currentAbbreviationProperty().set(abbreviation2);
    viewModel.deleteAbbreviation();

    assertNull(viewModel.currentAbbreviationProperty().get());
    assertFalse(viewModel.abbreviationsProperty().contains(abbreviation2));
}
```

< Provide a screenshot of the old coverage results (the same as you already showed above) >

Element	Missed Instructions	Cov.	Missed Branches	Cov.	Missed	Cxty	Missed	Lines	Missed	Methods
deleteAbbreviation()		64%		37%	4	5	3	10	0	1

< Provide a screenshot of the new coverage results >

deleteAbbreviation()		100%		100%	0	5	0	15	0	1

< State the coverage improvement with a number and elaborate on why the coverage is improved >

The coverage improved by 63%.

In the existing test(line 580), for the deleteAbbreviation method, the following branches are covered: currentAbbreviation != null, !currentAbbreviation.get().isPseudoAbbreviation(), and index > 1. This scenario tested includes setting a valid non-pseudo abbreviation and checking the deletion process when the abbreviation index is greater than 1.
The original tests are intertwined with other parts, making modifications cumbersome.

Therefore, I have rewritten the test functions for all branches: Current Abbreviation is null,Current Abbreviation is pseudo,Current Abbreviation is not null nor pseudo, with index >1, <=1,or index + 1 <abbreviationsCount.get(),index + 1 >= abbreviationsCount.get()

Overall

< Provide a screenshot of the old coverage results by running an existing tool (the same as you already showed above) >

org.jabref.logic.undo		0%		n/a	7	7	14	14	7	7	3	3
org.jabref.logic.shared.event		0%		n/a	8	8	16	16	8	8	3	3
org.jabref.model.search		50%		0%	5	11	8	18	3	9	0	1
org.jabref.logic		43%		50%	7	12	13	22	6	11	3	5
org.jabref.model.search.matchers		75%		70%	6	21	8	31	3	16	0	6
org.jabref.logic.remote		86%		93%	3	25	11	66	2	17	0	4
org.jabref.logic.shared.exception		0%		n/a	7	7	14	14	7	7	4	4
org.jabref.model.entry.event		84%		71%	7	25	10	55	3	18	0	5
org.jabref.logic.quality.consistency		96%		73%	7	61	6	171	2	48	0	6
org.jabref.gui.mergeentries.newmergedialog.fieldsmerger		80%		73%	8	27	8	43	2	12	1	5
org.jabref.logic.util.strings		99%		84%	11	66	3	469	1	28	0	10
org.jabref.logic.logging		0%		n/a	6	6	10	10	6	6	1	1
org.jabref.model.openoffice		68%		75%	3	12	3	20	2	10	1	2
org.jabref.logic.texparser		94%		85%	4	36	9	91	0	22	0	2
org.jabref.logic.remote.client		83%		50%	4	9	4	27	0	5	0	1
org.jabref.logic.shared.security		77%		n/a	1	5	2	15	1	5	0	1
org.jabref.gui.logging		0%		n/a	6	6	9	9	6	6	1	1
org.jabref.gui.util.comparator		90%		82%	4	23	5	47	0	6	0	3
org.jabref.logic.auxparser		97%		91%	3	41	3	107	0	24	0	2
org.jabref.model.database.event		82%		n/a	3	11	5	22	3	11	1	5
org.jabref.logic.preview		0%		n/a	1	1	1	1	1	1	1	1
org.jabref.logic.pseudonymization		97%		83%	2	18	3	43	1	15	0	3
org.jabref.model.schema		90%		100%	0	8	3	23	0	5	0	1
org.jabref.logic.push		97%		70%	3	9	1	16	0	4	0	1
org.jabref.model.metadata.event		66%		n/a	1	2	1	4	1	2	0	1
org.jabref.model.groups.event		66%		n/a	1	2	1	4	1	2	0	1
org.jabref.http		0%		n/a	1	1	1	1	1	1	1	1
org.jabref.logic.formatter		100%		100%	0	22	0	34	0	14	0	2
org.jabref.logic.help		100%		n/a	0	3	0	44	0	3	0	1
org.jabref.logic.formatter.minifier		100%		100%	0	20	0	31	0	13	0	2
org.jabref.logic.importer.fileformat.medline		100%		n/a	0	5	0	5	0	5	0	5
org.jabref.logic.importer.fileformat.mods		100%		n/a	0	5	0	6	0	5	0	3
org.jabref.model.paging		100%		n/a	0	6	0	11	0	6	0	1
org.jabref.logic.groups		100%		n/a	0	1	0	3	0	1	0	1
Total	170,530 of 327,443	47%	10,908 of 22,031	50%	15,642	26,144	36,689	66,723	8,757	14,663	781	1,784

< Provide a screenshot of the new coverage results by running the existing tool using all test modifications made by the group >

org.jabref.gui.collab.entrydelete		0%		n/a	4	4	10	10	4	4	1	1
org.jabref.gui.collab.entryadd		0%		n/a	4	4	10	10	4	4	1	1
org.jabref.model.strings		98%		86%	32	188	5	524	2	52	0	3
org.jabref.logic.formatter.casechanger		96%		88%	7	115	7	202	1	70	0	10
org.jabref.logic.remote.server		80%		71%	6	29	15	72	0	18	0	3
org.jabref.logic.undo		0%		n/a	7	7	14	14	7	7	3	3
org.jabref.logic.shared.event		0%		n/a	8	8	16	16	8	8	3	3
org.jabref.model.search		50%		0%	5	11	8	18	3	9	0	1
org.jabref.model.search.matchers		75%		70%	6	21	8	31	3	16	0	6
org.jabref.logic.remote		86%		93%	3	25	11	66	2	17	0	4
org.jabref.logic.shared.exception		0%		n/a	7	7	14	14	7	7	4	4
org.jabref.model.entry.event		84%		71%	7	25	10	55	3	18	0	5
org.jabref.logic.quality.consistency		96%		73%	7	61	6	171	2	48	0	6
org.jabref.gui.mergeentries.newmergedialog.fieldsmerger		80%		73%	8	27	8	43	2	12	1	5
org.jabref.logic.util.strings		99%		84%	11	66	3	469	1	28	0	10
org.jabref.logic		55%		50%	6	12	10	22	5	11	3	5
org.jabref.logic.logging		0%		n/a	6	6	10	10	6	6	1	1
org.jabref.model.openoffice		68%		75%	3	12	3	20	2	10	1	2
org.jabref.logic.texparser		94%		85%	4	36	9	91	0	22	0	2
org.jabref.logic.remote.client		83%		50%	4	9	4	27	0	5	0	1
org.jabref.logic.shared.security		77%		n/a	1	5	2	15	1	5	0	1
org.jabref.gui.logging		0%		n/a	6	6	9	9	6	6	1	1
org.jabref.gui.util.comparator		90%		82%	4	23	5	47	0	6	0	3
org.jabref.logic.auxparser		97%		91%	3	41	3	107	0	24	0	2
org.jabref.model.database.event		82%		n/a	3	11	5	22	3	11	1	5
org.jabref.logic.preview		0%		n/a	1	1	1	1	1	1	1	1
org.jabref.logic.pseudonymization		97%		83%	2	18	3	43	1	15	0	3
org.jabref.model.schema		90%		100%	0	8	3	23	0	5	0	1
org.jabref.logic.push		97%		70%	3	9	1	16	0	4	0	1
org.jabref.model.metadata.event		66%		n/a	1	2	1	4	1	2	0	1
org.jabref.model.groups.event		66%		n/a	1	2	1	4	1	2	0	1
org.jabref.http		0%		n/a	1	1	1	1	1	1	1	1
org.jabref.logic.formatter		100%		100%	0	22	0	34	0	14	0	2
org.jabref.logic.help		100%		n/a	0	3	0	44	0	3	0	1
org.jabref.logic.formatter.minifier		100%		100%	0	20	0	31	0	13	0	2
org.jabref.logic.importer.fileformat.medline		100%		n/a	0	5	0	5	0	5	0	5
org.jabref.logic.importer.fileformat.mods		100%		n/a	0	5	0	6	0	5	0	3
org.jabref.model.paging		100%		n/a	0	6	0	11	0	6	0	1
org.jabref.logic.groups		100%		n/a	0	1	0	3	0	1	0	1
Total	169,796 of 327,904	48%	10,847 of 22,059	50%	15,585	26,173	36,545	66,808	8,733	14,678	781	1,784

The coverage has improved by 1%, from 47% to 48%.

Statement of individual contributions

<Write what each group member did>

Xingyun Wang:
Instrument and improve code coverage of equals() and get().

Yiyang Sun:
Instrument and improve code coverage of getMainFileDirectory()
and getWriteMetadatatoPdf().

Yuli Wang:
Instrument and improve code coverage of deleteAbbreviation() and boolean equals(Object o).