

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	Ctxy	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.context	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.commoncontrols	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%			15%	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.fileannotationstab	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

Element	U+0	I+0	I+1	I+2	I+3	I+4	I+5	I+6	I+7	I+8	I+9	I+10
org.jabref.logic.shared.event	0%	n/a	8	8	16	16	8	8	8	3	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.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	1	1	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>

```
Function equals(Object o) instrumented code #1
Changes from all commits ▾ File filter ▾ Conversations ▾ Jump to ▾ ⚙ ▾
src/main/java/org/jabref/model/groups/ExplicitGroup.java ▾

28 39     public void addLegacyEntryKey(String key) {
29 40         this.legacyEntryKeys.add(key);
30 41     }

@@ -39,21 +50,34 @@ public AbstractGroup DeepCopy() {
39 50     @Override
40 51     public boolean equals(Object o) {
41 52         if (this == o) {
42 53         +             branchCoverage.put("equals_branch_1", true);
43 54             return true;
44 55         }
45 56         if (!(o instanceof ExplicitGroup)) {
46 57         +             branchCoverage.put("equals_branch_2", true);
47 58             return false;
48 59         }
49 60         ExplicitGroup other = (ExplicitGroup) o;
50 61         -             return Objects.equals(getName(), other.getName());
51 62         +             boolean result = Objects.equals(getName(), other.getName())
52 63             && Objects.equals(getHierarchicalContext(), other.getHierarchicalContext())
53 64             && Objects.equals(getIconName(), other.getIconName())
54 65             && Objects.equals(getDescription(), other.getDescription())
55 66             && Objects.equals(getColor(), other.getColor())
56 67             && Objects.equals(isExpanded(), other.isExpanded())
57 68             && Objects.equals(getLegacyEntryKeys(), other.getLegacyEntryKeys());
58 69         +             branchCoverage.put("equals_branch_3", true);
59 70         +             return result;
60 71     }
61 72     +
62 73     +
63 74     public static void printCoverage(){}
```

< 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()

jabref > src > main > java > org > jabref > gui > keyboard > © KeyBindingRepository > ⏪ 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/0a7060ee314f2edd5230ad970bb4c4080432600>

7

```
1   1     package org.jabref.gui.keyboard;
2   2
3   3     - import java.util.ArrayList;
4   4     - import java.util.Arrays;
5   5     - import java.util.Collections;
6   6     - import java.util.Comparator;
7   7     - import java.util.List;
8   8     - import java.util.Optional;
9   9     - import java.util.Set;
10 10    - import java.util.SortedMap;
11 11    - import java.util.TreeMap;
12 12    + import java.util.*;
13 13    + import java.util.stream.Collectors;
14 14    + import javafx.scene.input.KeyCode;
15 15
16 16     @@ -33,6 +25,14 @ public KeyBindingRepository(SortedMap<KeyBinding, String> bindings) {
17 17         this.bindings = bindings;
18 18     }
19 19
20 20     +     public static Map<String,Boolean> branchCoverage = new HashMap<>();
21 21     +     static{
22 22         branchCoverage.put("equals_branch_1",false);
23 23         branchCoverage.put("equals_branch_2",false);
24 24         branchCoverage.put("equals_branch_3",false);
25 25     }
26 26
27 27
28 28     +     public KeyBindingRepository(List<String> bindNames, List<String> bindings) {
29 29         this.bindings = new TreeMap<>(Comparator.comparing(KeyBinding::getLocalization));
30 30     }
31 31
32 32
33 33
34 34
35 35
36 36
37 37
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,
49
50     @@ -77,12 +81,25 @@ public StringProperty getUserAndHostProperty() {
51
52
53
54
55
56
57
58
59
59
60
61
62
63
64
65
66
67
68
69
69
70
71
72
73
74
75
76
77
78
79
80
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,
11 +         null, Set.of(), false, null, false, false);
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 >



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/b9473f6448abe68e9aa53c67b50c597f6ad4517>

5

```
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

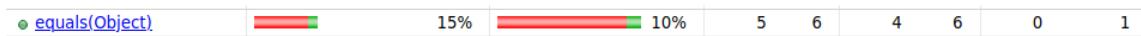
Check printout info:

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 >

```

v 24 src/main/java/org/jabref/gui/theme/Theme.java ...
+ ... @@ -2,6 +2,8 @@
 2   2
 3   3     import java.util.Objects;
 4   4     import java.util.Optional;
 5 + 5     + import java.util.HashMap;
 6 + 6     + import java.util.Map;
 7   7
 8   8     /**
 9      * Represents one of three types of a css based Theme for JabRef:
10
11
12 11 12     @@ -22,6 +24,13 @@ public enum Type {
13 13     private final String name;
14 14     private final Optional<StyleSheet> additionalStylesheet;
15
16 15     + public static Map<String, Boolean> branchCoverage = new HashMap<>();
17 16     static {
18 17         branchCoverage.put("this == o", false);
19 18         branchCoverage.put("o == null", false);
20 19         branchCoverage.put("getClass() != o.getClass()", false);
21 20     }
22 21
23 22     public Theme(String name) {
24 23         Objects.requireNonNull(name);
25 24
26 25     @@ -95,9 +104,15 @@ public Optional<StyleSheet> getAdditionalStylesheet() {
27 26
28 27     @Override
29 28     public boolean equals(Object o) {
30 29         if (this == o) {
31 30             branchCoverage.put("this == o", true);
32 31         return true;
33 32     }
34
35     @@ -104,9 +104,15 @@ public boolean equals(Object o) {
36 35         if (this == o) {
37 36             branchCoverage.put("this == o", true);
38 37         return true;
39 38     }
40 39     if (o == null || getClass() != o.getClass()) {
41 40         if (o == null) {
42 41             branchCoverage.put("o == null", true);
43 42         return false;
44 43     }
45 44     if (getClass() != o.getClass()) {
46 45         branchCoverage.put("getClass() != o.getClass()", true);
47 46     return false;
48 47 }
49 48     Theme that = (Theme) o;
50
51 50     @@ -116,4 +131,11 @@ public String toString() {
52 51         ", name='" + name + '\'' +
53 52         '}';
54 53
55 54     }
56
57 55
58 56     + public static void printCoverage() {
59 57     for (Map.Entry<String, Boolean> entry : branchCoverage.entrySet()) {
60 58         System.out.println("Branch " + entry.getKey() + " was " + (entry.getValue() ? "hit" : "not hit"));
61 59     }
62 60
63 61 }
64
65 62
66 63
67 64
68 65
69 66
70 67
71 68
72 69
73 70
74 71
75 72
76 73
77 74
78 75
79 76
80 77
81 78
82 79
83 80
84 81
85 82
86 83
87 84
88 85
89 86
90 87
91 88
92 89
93 90
94 91
95 92
96 93
97 94
98 95
99 96
100 97
101 98
102 99
103 100
104 101
105 102
106 103
107 104
108 105
109 106
110 107
111 108
112 109
113 110
114 111
115 112
116 113
117 114
118 115
119 116
120 117
121 118
122 119
123 120
124 121
125 122
126 123
127 124
128 125
129 126
130 127
131 128
132 129
133 130
134 131
135 132
136 133
137 134
138 135
139 136
140 137
141 138
142 139
143 140
144 141
145 142
146 143
147 144
148 145
149 146
150 147
151 148
152 149
153 150
154 151
155 152
156 153
157 154
158 155
159 156
160 157
161 158
162 159
163 160
164 161
165 162
166 163
167 164
168 165
169 166
170 167
171 168
172 169
173 170
174 171
175 172
176 173
177 174
178 175
179 176
180 177
181 178
182 179
183 180
184 181
185 182
186 183
187 184
188 185
189 186
190 187
191 188
192 189
193 190
194 191
195 192
196 193
197 194
198 195
199 196
200 197
201 198
202 199
203 200
204 201
205 202
206 203
207 204
208 205
209 206
210 207
211 208
212 209
213 210
214 211
215 212
216 213
217 214
218 215
219 216
220 217
221 218
222 219
223 220
224 221
225 222
226 223
227 224
228 225
229 226
230 227
231 228
232 229
233 230
234 231
235 232
236 233
237 234
238 235
239 236
240 237
241 238
242 239
243 240
244 241
245 242
246 243
247 244
248 245
249 246
250 247
251 248
252 249
253 250
254 251
255 252
256 253
257 254
258 255
259 256
260 257
261 258
262 259
263 260
264 261
265 262
266 263
267 264
268 265
269 266
270 267
271 268
272 269
273 270
274 271
275 272
276 273
277 274
278 275
279 276
280 277
281 278
282 279
283 280
284 281
285 282
286 283
287 284
288 285
289 286
290 287
291 288
292 289
293 290
294 291
295 292
296 293
297 294
298 295
299 296
300 297
301 298
302 299
303 300
304 301
305 302
306 303
307 304
308 305
309 306
310 307
311 308
312 309
313 310
314 311
315 312
316 313
317 314
318 315
319 316
320 317
321 318
322 319
323 320
324 321
325 322
326 323
327 324
328 325
329 326
330 327
331 328
332 329
333 330
334 331
335 332
336 333
337 334
338 335
339 336
340 337
341 338
342 339
343 340
344 341
345 342
346 343
347 344
348 345
349 346
350 347
351 348
352 349
353 350
354 351
355 352
356 353
357 354
358 355
359 356
360 357
361 358
362 359
363 360
364 361
365 362
366 363
367 364
368 365
369 366
370 367
371 368
372 369
373 370
374 371
375 372
376 373
377 374
378 375
379 376
380 377
381 378
382 379
383 380
384 381
385 382
386 383
387 384
388 385
389 386
390 387
391 388
392 389
393 390
394 391
395 392
396 393
397 394
398 395
399 396
400 397
401 398
402 399
403 400
404 401
405 402
406 403
407 404
408 405
409 406
410 407
411 408
412 409
413 410
414 411
415 412
416 413
417 414
418 415
419 416
420 417
421 418
422 419
423 420
424 421
425 422
426 423
427 424
428 425
429 426
430 427
431 428
432 429
433 430
434 431
435 432
436 433
437 434
438 435
439 436
440 437
441 438
442 439
443 440
444 441
445 442
446 443
447 444
448 445
449 446
450 447
451 448
452 449
453 450
454 451
455 452
456 453
457 454
458 455
459 456
460 457
461 458
462 459
463 460
464 461
465 462
466 463
467 464
468 465
469 466
470 467
471 468
472 469
473 470
474 471
475 472
476 473
477 474
478 475
479 476
480 477
481 478
482 479
483 480
484 481
485 482
486 483
487 484
488 485
489 486
490 487
491 488
492 489
493 490
494 491
495 492
496 493
497 494
498 495
499 496
500 497
501 498
502 499
503 500
504 501
505 502
506 503
507 504
508 505
509 506
510 507
511 508
512 509
513 510
514 511
515 512
516 513
517 514
518 515
519 516
520 517
521 518
522 519
523 520
524 521
525 522
526 523
527 524
528 525
529 526
530 527
531 528
532 529
533 530
534 531
535 532
536 533
537 534
538 535
539 536
540 537
541 538
542 539
543 540
544 541
545 542
546 543
547 544
548 545
549 546
550 547
551 548
552 549
553 550
554 551
555 552
556 553
557 554
558 555
559 556
560 557
561 558
562 559
563 560
564 561
565 562
566 563
567 564
568 565
569 566
570 567
571 568
572 569
573 570
574 571
575 572
576 573
577 574
578 575
579 576
580 577
581 578
582 579
583 580
584 581
585 582
586 583
587 584
588 585
589 586
590 587
591 588
592 589
593 590
594 591
595 592
596 593
597 594
598 595
599 596
599 599
600 600
601 601
602 602
603 603
604 604
605 605
606 606
607 607
608 608
609 609
610 610
611 611
612 612
613 613
614 614
615 615
616 616
617 617
618 618
619 619
620 620
621 621
622 622
623 623
624 624
625 625
626 626
627 627
628 628
629 629
630 630
631 631
632 632
633 633
634 634
635 635
636 636
637 637
638 638
639 639
640 640
641 641
642 642
643 643
644 644
645 645
646 646
647 647
648 648
649 649
650 650
651 651
652 652
653 653
654 654
655 655
656 656
657 657
658 658
659 659
660 660
661 661
662 662
663 663
664 664
665 665
666 666
667 667
668 668
669 669
670 670
671 671
672 672
673 673
674 674
675 675
676 676
677 677
678 678
679 679
680 680
681 681
682 682
683 683
684 684
685 685
686 686
687 687
688 688
689 689
690 690
691 691
692 692
693 693
694 694
695 695
696 696
697 697
698 698
699 699
700 700
701 701
702 702
703 703
704 704
705 705
706 706
707 707
708 708
709 709
710 710
711 711
712 712
713 713
714 714
715 715
716 716
717 717
718 718
719 719
720 720
721 721
722 722
723 723
724 724
725 725
726 726
727 727
728 728
729 729
730 730
731 731
732 732
733 733
734 734
735 735
736 736
737 737
738 738
739 739
740 740
741 741
742 742
743 743
744 744
745 745
746 746
747 747
748 748
749 749
750 750
751 751
752 752
753 753
754 754
755 755
756 756
757 757
758 758
759 759
760 760
761 761
762 762
763 763
764 764
765 765
766 766
767 767
768 768
769 769
770 770
771 771
772 772
773 773
774 774
775 775
776 776
777 777
778 778
779 779
780 780
781 781
782 782
783 783
784 784
785 785
786 786
787 787
788 788
789 789
790 790
791 791
792 792
793 793
794 794
795 795
796 796
797 797
798 798
799 799
800 800
801 801
802 802
803 803
804 804
805 805
806 806
807 807
808 808
809 809
810 810
811 811
812 812
813 813
814 814
815 815
816 816
817 817
818 818
819 819
820 820
821 821
822 822
823 823
824 824
825 825
826 826
827 827
828 828
829 829
830 830
831 831
832 832
833 833
834 834
835 835
836 836
837 837
838 838
839 839
840 840
841 841
842 842
843 843
844 844
845 845
846 846
847 847
848 848
849 849
850 850
851 851
852 852
853 853
854 854
855 855
856 856
857 857
858 858
859 859
860 860
861 861
862 862
863 863
864 864
865 865
866 866
867 867
868 868
869 869
870 870
871 871
872 872
873 873
874 874
875 875
876 876
877 877
878 878
879 879
880 880
881 881
882 882
883 883
884 884
885 885
886 886
887 887
888 888
889 889
890 890
891 891
892 892
893 893
894 894
895 895
896 896
897 897
898 898
899 899
900 900
901 901
902 902
903 903
904 904
905 905
906 906
907 907
908 908
909 909
910 910
911 911
912 912
913 913
914 914
915 915
916 916
917 917
918 918
919 919
920 920
921 921
922 922
923 923
924 924
925 925
926 926
927 927
928 928
929 929
930 930
931 931
932 932
933 933
934 934
935 935
936 936
937 937
938 938
939 939
940 940
941 941
942 942
943 943
944 944
945 945
946 946
947 947
948 948
949 949
950 950
951 951
952 952
953 953
954 954
955 955
956 956
957 957
958 958
959 959
960 960
961 961
962 962
963 963
964 964
965 965
966 966
967 967
968 968
969 969
970 970
971 971
972 972
973 973
974 974
975 975
976 976
977 977
978 978
979 979
980 980
981 981
982 982
983 983
984 984
985 985
986 986
987 987
988 988
989 989
990 990
991 991
992 992
993 993
994 994
995 995
996 996
997 997
998 998
999 999
999 999

```

```

v 7 src/test/java/org/jabref/gui/theme/ThemeTest.java ...
+ ... @@ -2,12 +2,19 @@
 2   2
 3   3     import java.util.Optional;
 4
 5 + 5     + import org.junit.jupiter.api.AfterAll;
 6  6     import org.junit.jupiter.api.Test;
 7
 8  7     import static org.junit.jupiter.api.Assertions.assertEquals;
 9  8     import static org.junit.jupiter.api.Assertions.assertTrue;
10  9
11 10     public class ThemeTest {
12 11
13 12     + @AfterAll
14 13     public static void printCoverage() {
15 14         Theme.printCoverage();
16 15     }
17 16
18 17     @Test
19 18     public void lightThemeUsedWhenPathIsBlank() {
20 19         Theme theme = new Theme("");
21 20
22 21
23 22
24 23
25 24
26 25
27 26
28 27
29 28
30 29
31 30
32 31
33 32
34 33
35 34
36 35
37 36
38 37
39 38
39 39
40 40
41 41
42 42
43 43
44 44
45 45
46 46
47 47
48 48
49 49
50 50
51 51
52 52
53 53
54 54
55 55
56 56
57 57
58 58
59 59
59 59
60 60
61 61
62 62
63 63
64 64
65 65
66 66
67 67
68 68
69 69
70 70
71 71
72 72
73 73
74 74
75 75
76 76
77 77
78 78
79 79
80 80
81 81
82 82
83 83
84 84
85 85
86 86
87 87
88 88
89 89
90 90
91 91
92 92
93 93
94 94
95 95
96 96
97 97
98 98
99 99
99 99

```

< 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

```

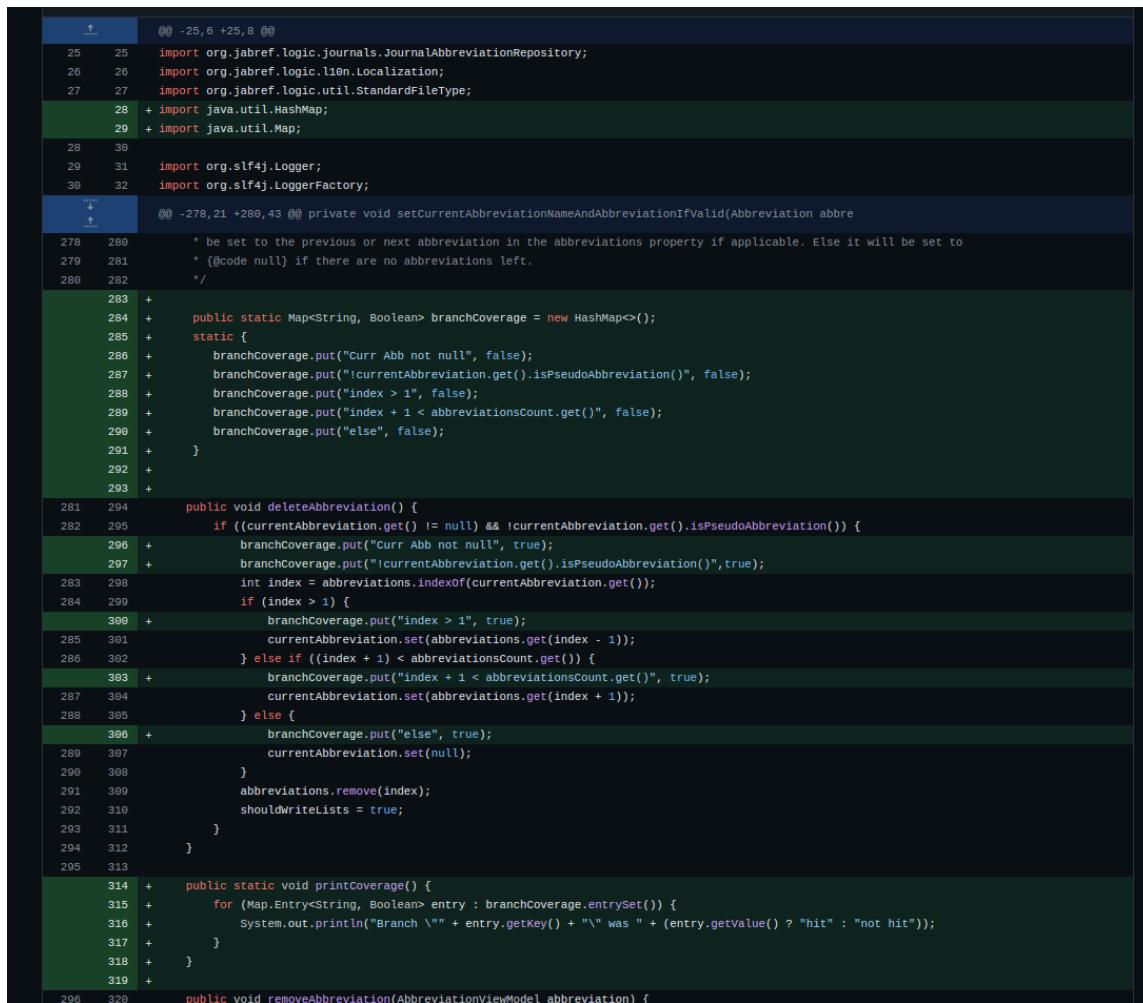
< Function 2 name >

```

void deleteAbbreviation()
src/main/java/org/jabref/gui/preferences/journals/JournalAbbreviationsTabViewModel.java
a-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 + 28 + import java.util.HashMap;
29 + 29 + import java.util.Map;
30   30
31   31 import org.slf4j.Logger;
32   32 import org.slf4j.LoggerFactory;
@@ -278,21 +280,43 @@
300 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 +
286 +     static {
287 +         branchCoverage.put("Curr Abb not null", false);
288 +         branchCoverage.put("!currentAbbreviation.get().isPseudoAbbreviation()", false);
289 +         branchCoverage.put("index > 1", false);
290 +         branchCoverage.put("index + 1 < abbreviationsCount.get()", false);
291 +         branchCoverage.put("else", false);
292 +
293 +
294     public void deleteAbbreviation() {
295         if ((currentAbbreviation.get() != null) && !currentAbbreviation.get().isPseudoAbbreviation()) {
296             branchCoverage.put("curr Abb not null", true);
297             branchCoverage.put("!currentAbbreviation.get().isPseudoAbbreviation()", true);
298             int index = abbreviations.indexOf(currentAbbreviation.get());
299             if (index > 1) {
300                 branchCoverage.put("index > 1", true);
301                 currentAbbreviation.set(abbreviations.get(index - 1));
302             } else if ((index + 1) < abbreviationsCount.get()) {
303                 branchCoverage.put("index + 1 < abbreviationsCount.get()", true);
304                 currentAbbreviation.set(abbreviations.get(index + 1));
305             } else {
306                 branchCoverage.put("else", true);
307                 currentAbbreviation.set(null);
308             }
309             abbreviations.remove(index);
310             shouldWriteLists = true;
311         }
312     }
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 +
320     public void removeAbbreviation(abbreviationViewModel abbreviation) {

```

```

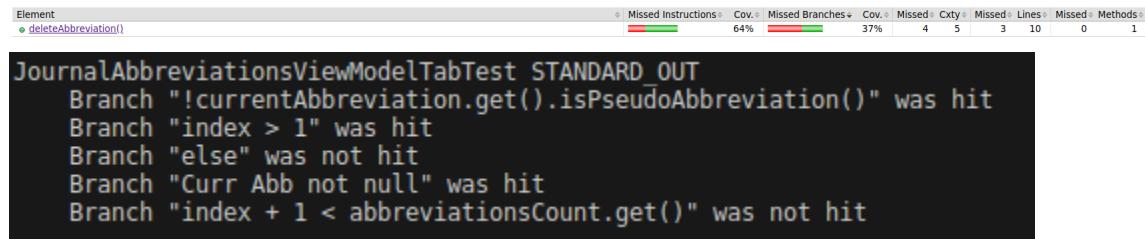
@@ -20,7 +20,7 @@
20 20 import org.jabref.logic.journals.JournalAbbreviationLoader;
21 21 import org.jabref.logic.journals.JournalAbbreviationPreferences;
22 22 import org.jabref.logic.journals.JournalAbbreviationRepository;
23 23 + import org.junit.jupiter.api.AfterAll;
24 24 import org.junit.jupiter.api.BeforeEach;
25 25 import org.junit.jupiter.api.Test;
26 26 import org.junit.jupiter.api.io.TempDir;
@@ -187,6 +187,12 @@ void setUpViewModel(@TempDir Path tempFolder) throws Exception {
187 187     emptyTestFile = createTestFile(new CsvFileNameAndContent("emptyTestFile.csv", ""));
188 188 }
189 189
190 190 + @AfterAll
191 191 + public static void printCoverage() {
192 192 +     JournalAbbreviationsTabViewModel.printCoverage();
193 193 }
194 194 +
195 195 +
196 196 @Test
197 197 void initialHasNoFilesAndNoAbbreviations() {
198 198     assertEquals(0, viewModel.journalFilesProperty().size());

```

< 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/0a7060ee314f2edd5230ad970bb4c4080432600>

7

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

```

@@ -16,13 +16,14 @@ class ExplicitGroupTest {
    16   16
    17   17     private ExplicitGroup group;
    18   18     private ExplicitGroup group2;
  19 -  +
    19 +   private ExplicitGroup group3;
    20   20     private BibEntry entry;
    21   21
    22   22     @BeforeEach
    23   23     void setUp() {
    24   24       group = new ExplicitGroup("myExplicitGroup", GroupHierarchyType.INDEPENDENT, ',');
  25 -  -
    25 +   group2 = new ExplicitGroup("myExplicitGroup2", GroupHierarchyType.INCLUDING, ',');
    26 +   group3 = new ExplicitGroup("myExplicitGroup", GroupHierarchyType.INDEPENDENT, ','); // Same as group
    26 +   group3 = new ExplicitGroup("anotherGroup", GroupHierarchyType.INDEPENDENT, ','); // Different from group
    27   28     entry = new BibEntry();
    28   29   }
@@ -40,10 +41,10 @@ void addSingleGroupToNonemptyBibEntryAppendsToGroupsField() {
    40   41   }
    41   42
    42   43     @Test
  43 -  -
    43 +   void addTwoGroupsToBibEntryChangesGroupsField() {
    44 -  -
    44 +   group.add(entry);
    45 -  -
    45 +   group2.add(entry);
  46 -  -
    46 +   assertEquals(Optional.of("myExplicitGroup, myExplicitGroup2"), entry.getField(StandardField.GROUPS));
    44 +   void addTwoGroupsToBibEntryChangesGroupsField() {
    45 +   group.add(entry);
    46 +   group2.add(entry);
    47 +   assertEquals(Optional.of("myExplicitGroup, myExplicitGroup2"), entry.getField(StandardField.GROUPS));
    47   48   }
    48   49

```

< 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 45%

```

    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/e01d36b7c093e79f92eb839e238c31636eba3b9>

c

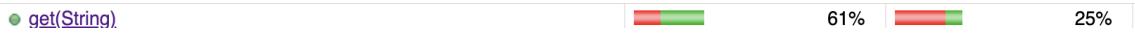
See changes in src/test/java/org/jabref/gui/keyboard/KeyBindingsTabModelTest.java

```

@@ -16,13 +16,14 @@ class ExplicitGroupTest {
    16   16
    17   17     private ExplicitGroup group;
    18   18     private ExplicitGroup group2;
  19 - 
  19 +     private ExplicitGroup group3;
    20  20     private BibEntry entry;
    21  21
    22  22     @BeforeEach
    23  23     void setUp() {
    24  24         group = new ExplicitGroup("myExplicitGroup", GroupHierarchyType.INDEPENDENT, ',');
  25 -         group2 = new ExplicitGroup("myExplicitGroup2", GroupHierarchyType.INCLUDING, ',');
  25 +         group2 = new ExplicitGroup("myExplicitGroup", GroupHierarchyType.INDEPENDENT, ','); // Same as group
  26 +         group3 = new ExplicitGroup("anotherGroup", GroupHierarchyType.INDEPENDENT, ','); // Different from group
    26  26         entry = new BibEntry();
    27  27     }
    28  28
    29  29
@@ -40,10 +41,10 @@ void addSingleGroupToNonemptyBibEntryAppendsToGroupsField() {
    40  41     }
    41  42
    42  43     @Test
  43 -     void addTwoGroupsToBibEntryChangesGroupsField() {
  44 -         group.add(entry);
  45 -         group2.add(entry);
  46 -         assertEquals(Optional.of("myExplicitGroup, myExplicitGroup2"), entry.getField(StandardField.GROUPS));
  44 +     void addTwoGroupsToBibEntryChangesGroupsField() {
  45 +         group.add(entry);
  46 +         group2.add(entry);
  47 +         assertEquals(Optional.of("myExplicitGroup, myExplicitGroup2"), entry.getField(StandardField.GROUPS));
    47  48     }
    48  49

```

< 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%

```

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";
    }
}

```

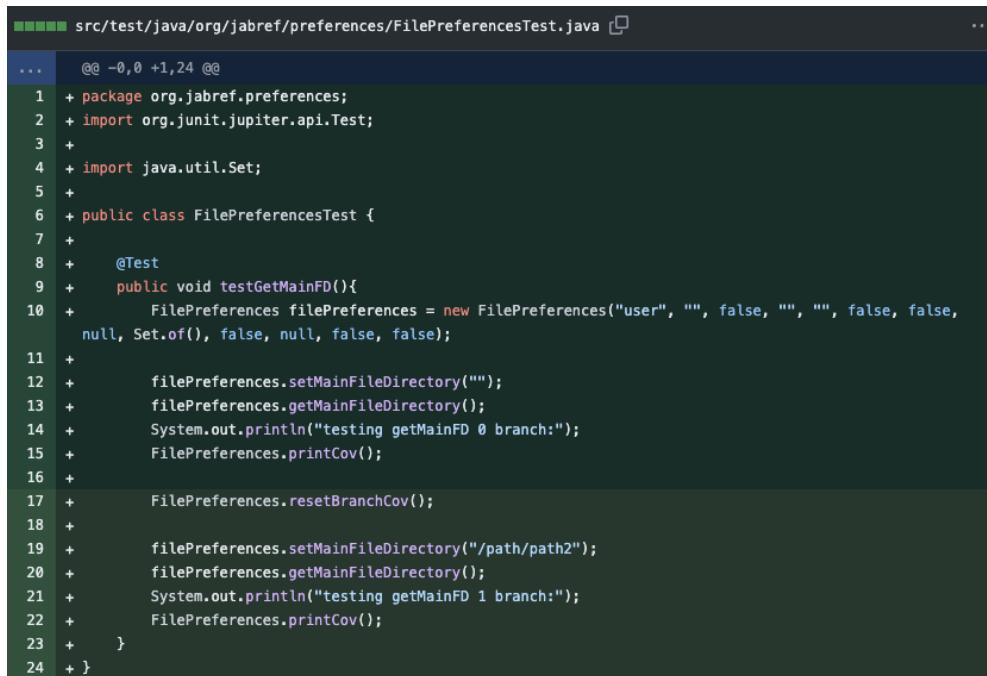
The console messages indicate that only the first branch is being hit. To cover all branches, I have added two test cases: one for handling the case where the result is null, and another for when there is no key binding.

< 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 @@
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,
11+         null, Set.of(), false, null, false, false);
12+
13+         filePreferences.setMainFileDirectory("");
14+         filePreferences.getMainFileDirectory();
15+         System.out.println("testing getMainFD 0 branch:");
16+         filePreferences.printCov();
17+
18+         filePreferences.resetBranchCov();
19+
20+         filePreferences.setMainFileDirectory("/path/path2");
21+         filePreferences.getMainFileDirectory();
22+         System.out.println("testing getMainFD 1 branch:");
23+         filePreferences.printCov();
24+     }
25+ }
```

< 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_EMBEDDIBFILEINPDF() {
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.euquals

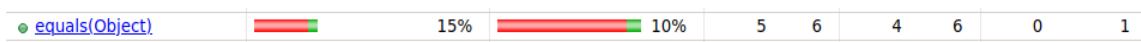
< Show a patch (diff) or a link to a commit made in your forked repository that shows the new/enhanced test >

```

63     /1           assertEquals( "donotexist.css" , theme.getAdditionalStyleSheets().get().getWatchPath().getFilename() );
64     }
65
66     +
67     @Test
68     public void thisEquals0() {
69         Theme theme = new Theme("test");
70         assertTrue(theme.equals(theme));
71     }
72
73     +
74     @Test
75     public void oEqualsNull() {
76         Theme theme = new Theme("test");
77         assertFalse(theme.equals(null));
78     }
79
80     +
81     @Test
82     public void differentClass() {
83         Theme theme = new Theme("test");
84         String differentClassObject = "test";
85         assertFalse(theme.equals(differentClassObject));
86     }
87
88     +
89     @Test
90     public void differentType() {
91         Theme theme1 = new Theme("test1");
92         Theme theme2 = new Theme("test2");
93         assertFalse(theme1.equals(theme2));
94     }
95
96     +
97     @Test
98     public void sameTypeAndName() {
99         Theme theme1 = new Theme("test");
100        Theme theme2 = new Theme("test");
101        assertTrue(theme1.equals(theme2));
102    }
103
104    +
105    @Test
106    public void sameAttributesDifferentObjects() {
107        Theme theme1 = new Theme("test");
108        Theme theme2 = new Theme("test");
109        assertTrue(theme1.equals(theme2));
110    }
111
112    +
113    @Test
114    public void sameTypeDifferentName() {
115        Theme theme1 = new Theme("test1");
116        Theme theme2 = new Theme("test2");
117        assertFalse(theme1.equals(theme2));
118    }
119
120    +
121 }

```

< 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 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 same & 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));
}
```

```

// 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 testDelAbbCurIsNotNulPseudoIndexMoreAbbCount() {
    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) >



< 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 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) >

	U %	lva	/	/	1+	1+	/	/	o	o
org.jabref.logic.univ	0%	n/a	8	8	16	16	8	8	3	3
org.jabref.logic.shared.event	50%	0%	5	11	8	18	3	9	0	1
org.jabref.model.search	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).