

New Test Cases

In the previous assignment, we fixed an issue where a warning was displayed in the entry editor when the title content is made up of two sentences ([PR #6054](#)). In this change, we modified the existing algorithm in `TitleChecker` to ensure that the title checking process makes sense for title content which is made up of one sentence/phrase or two sentences/phrases. There exists a test file called `IntegrityCheckTest` in which the functionality of `TitleChecker` gets checked in `testTitleChecks()`. However, this is more of end-to-end testing, and there is no dedicated pre-existing test class for `TitleChecker`. Hence, based upon suggestions from JabRef reviewers, we created a new test file called `TitleCheckerTest` and added test cases for titles having delimiters in various scenarios. Given below is the actual implementation.

Code Snippet (1) : TitleCheckerTest Class (newly added)

```
public class TitleCheckerTest {
    private TitleChecker checker;

    @BeforeEach
    public void setUp() {
        BibDatabaseContext databaseContext = new BibDatabaseContext();
        databaseContext.setMode(BibDatabaseMode.BIBTEX);
        checker = new TitleChecker(databaseContext);
    }

    @Test
    public void FirstLetterAsOnlyCapitalLetterInSubTitle2() {
        assertEquals(Optional.empty(), checker.checkValue("This is a sub title 1:
This is a sub title 2"));
    }

    @Test
    public void NoCapitalLetterInSubTitle2() {
        assertEquals(Optional.empty(), checker.checkValue("This is a sub title 1:
this is a sub title 2"));
    }

    @Test
    public void TwoCapitalLettersInSubTitle2() {
        assertNotEquals(Optional.empty(), checker.checkValue("This is a sub title 1:
This is A sub title 2"));
    }

    @Test
```

```
public void MiddleLetterAsOnlyCapitalLetterInSubTitle2() {
    assertNotEquals(Optional.empty(), checker.checkValue("This is a sub title 1:
this is A sub title 2"));
}

@Test
public void TwoCapitalLettersInSubTitle2WithCurlyBrackets() {
    assertEquals(Optional.empty(), checker.checkValue("This is a sub title 1:
This is {A} sub title 2"));
}

@Test
public void MiddleLetterAsOnlyCapitalLetterInSubTitle2WithCurlyBrackets() {
    assertEquals(Optional.empty(), checker.checkValue("This is a sub title 1:
this is {A} sub title 2"));
}

@Test
public void FirstLetterAsOnlyCapitalLetterInSubTitle2AfterContinuousDelimiters()
{
    assertEquals(Optional.empty(), checker.checkValue("This is a sub title
1...This is a sub title 2"));
}

@Test
public void
MiddleLetterAsOnlyCapitalLetterInSubTitle2AfterContinuousDelimiters() {
    assertNotEquals(Optional.empty(), checker.checkValue("This is a sub title
1... this is a sub Title 2"));
}

@Test
public void
FirstLetterAsOnlyCapitalLetterInEverySubTitleWithContinuousDelimiters() {
    assertEquals(Optional.empty(), checker.checkValue("This is; A sub title
1.... This is a sub title 2"));
}

@Test
public void FirstLetterAsOnlyCapitalLetterInEverySubTitleWithRandomDelimiters()
{
    assertEquals(Optional.empty(), checker.checkValue("This!is!!A!Title??"));
}

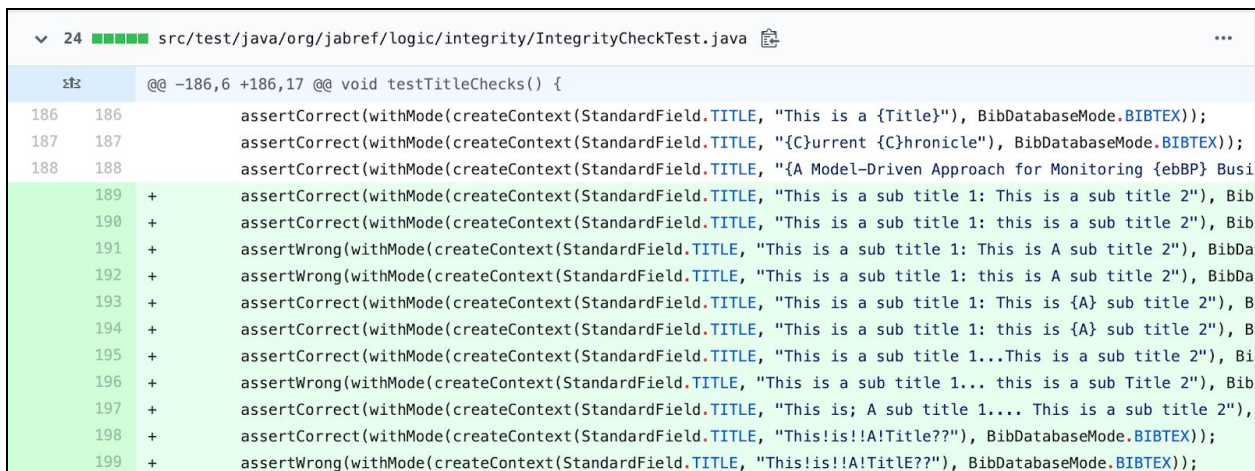
@Test
public void MoreThanOneCapitalLetterInSubTitleWithoutCurlyBrackets() {
    assertNotEquals(Optional.empty(), checker.checkValue("This!is!!A!Title??"));
}
```

```
}  
}
```

Background

In the contributing policy of JabRef, test cases are not required but are recommended to ensure that we solve the issue and hence, we do not break anything in the system. Initially, we **did not write** any formal test cases for the issue we were trying to fix. The only thing we did is to manually test various titles in the GUI application and run `gradlew check`, which is the minimum requirement to ensure that our changes did not break the system.

The reviewer gave us feedback that it would be better if we included the test cases. To respond, we were thinking of creating a dedicated test class for `TitleChecker` as the reviewer suggested. However, we found an existing test file called `IntegrityCheckTest`, inside of which there is a test method called `testTitleChecks` that have many assertions on different titles in either `BibDatabaseMode.BIBTEX` mode or `BibDatabaseMode.BIBLATEX` mode. As there exist methods for testing `TitleChecker`, it was unnecessary to create one. Hence, we **added more input values in the existing method** to show our modified `TitleChecker` works well.



```
24 src/test/java/org/jabref/logic/integrity/IntegrityCheckTest.java  
@@ -186,6 +186,17 @@ void testTitleChecks() {  
186 186      assertCorrect(withMode(createContext(StandardField.TITLE, "This is a {Title}"), BibDatabaseMode.BIBTEX));  
187 187      assertCorrect(withMode(createContext(StandardField.TITLE, "{C}urrent {C}hronicle"), BibDatabaseMode.BIBTEX));  
188 188      assertCorrect(withMode(createContext(StandardField.TITLE, "{A Model-Driven Approach for Monitoring {ebBP} Busi  
189 +      assertCorrect(withMode(createContext(StandardField.TITLE, "This is a sub title 1: This is a sub title 2"), Bib  
190 +      assertCorrect(withMode(createContext(StandardField.TITLE, "This is a sub title 1: this is a sub title 2"), Bib  
191 +      assertWrong(withMode(createContext(StandardField.TITLE, "This is a sub title 1: This is A sub title 2"), BibDa  
192 +      assertWrong(withMode(createContext(StandardField.TITLE, "This is a sub title 1: this is A sub title 2"), BibDa  
193 +      assertCorrect(withMode(createContext(StandardField.TITLE, "This is a sub title 1: This is {A} sub title 2"), B  
194 +      assertCorrect(withMode(createContext(StandardField.TITLE, "This is a sub title 1: this is {A} sub title 2"), B  
195 +      assertCorrect(withMode(createContext(StandardField.TITLE, "This is a sub title 1...This is a sub title 2"), Bi  
196 +      assertWrong(withMode(createContext(StandardField.TITLE, "This is a sub title 1... this is a sub Title 2"), Bib  
197 +      assertCorrect(withMode(createContext(StandardField.TITLE, "This is; A sub title 1... This is a sub title 2"),  
198 +      assertCorrect(withMode(createContext(StandardField.TITLE, "This!is!!A!TitlE??"), BibDatabaseMode.BIBTEX));  
199 +      assertWrong(withMode(createContext(StandardField.TITLE, "This!is!!A!TitlE??"), BibDatabaseMode.BIBTEX));
```

Figure (1): Additional inputs to check title

After submitting a new commit for the above changes in the existing test file, the reviewer pointed out that this is more of “end-to-end” testing from users’ perspective and it is not specific for `TitleChecker`. The reviewer pointed out that it was great to have new test cases for this method but their expectation is to have a new test class dedicated to test the functionality of `TitleChecker`. We realized that earlier we did not understand `IntegrityCheckTest` completely and learnt more about the concept of end-to-end testing.

Finally, we made another new commit **including a new test file** for `TitleCheckerTest`. In this test class, we set up the title checker before each test case and gave each test method a meaningful name (as noticed in Code Snippet (1)).

However, this is not the last commit as expected. The core developer explained that JabRef currently values a lot of the code quality and hence are in a “nitpicking” mode. We need to fix checkstyle issues in the recent commit. For example, there are more than one empty line and the order of imports is incorrect. We are planning to incorporate the new changes suggested and submit another commit. This was a great learning experience in contributing to the open source community and we are glad that we chose an active project with engaging developers!