UCI University of California, Irvine

New Test Cases for Omni-Notes

Dongxin Xiang, Jing Chen, Guowei Li

1 Introduction

There are many methods that are not covered by the existing test cases due to the limited amount of them. We picked three untested methods and created a test case for each of them. In this report, we will state these new test cases and our new understandings about Omni-Notes gained from the experience.

2 New Test Cases

2.1 getDateFromStringTest()

The first test case was built for the getDateFromString() method. The source code of getDateFromStringTest() is shown in Fig. 1. This method is used to convert a String object to a Calendar object according to a date format. It has two parameters: one is the original String (date), and the other one is the format (dateformat) used in the conversion, which is also a String object. In the test case (Fig. 2), we called the getDateFromString() method with input, a date string, and a date format string. Then we used assertEquals statements to check all the properties of the returned Calendar object. And the test case passed (Fig. 3).

```
public static Calendar getDateFromString (String str, String format) {
  Calendar cal = Calendar.getInstance();
  SimpleDateFormat sdf = new SimpleDateFormat(format);
  try {
    cal.setTime(sdf.parse(str));
  } catch (ParseException e) {
    LogDelegate.e("Malformed datetime string" + e.getMessage());

  } catch (NullPointerException e) {
    LogDelegate.e("Date or time not set");
  }
  return cal;
}
```

Fig.1 Source code of getDateFromString()

```
@Test
public void getDateFromStringTest() {
    String date="2020/03/12 15:15:15";
    String dateformat="yyyy/MM/dd HH:mm:ss";
    Calendar calendar=DateUtils.getDateFromString(date, dateformat);
    assertEquals( expected: 2020, calendar.get(Calendar.YEAR));
    assertEquals( expected: 2, calendar.get(Calendar.MONTH));
    assertEquals( expected: 12, calendar.get(Calendar.DATE));
    assertEquals( expected: 15, calendar.get(Calendar.HOUR_OF_DAY));
    assertEquals( expected: 15, calendar.get(Calendar.MINUTE));
    assertEquals( expected: 15, calendar.get(Calendar.SECOND));
}
```

Fig.2 Source code of getDataFromStringTest()

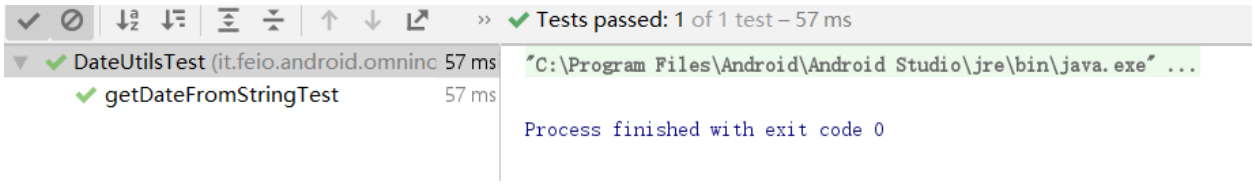

Fig. 3 Result of running getDataFromStringTest()

2.2 getLongFromDateTimeTest()

This test case was built for the getLongFromDateTime() method. The source code of getLongFromDateTime() method is shown in Fig. 4. Similar to the getDateFromString() method, getLongFromDateTime() is also used to convert a String object to a Calendar object. However, it requires separate formats and separate Strings for both date and time to build the different parts of the Calendar object. In the test case (Fig. 5), we called the getLongFromDateTimeTest() method with four strings as input including date, dateformat, time, and timeformat. Then we used assertEquals statements to check all the properties of the returned Calendar object. And the test case passed (Fig. 6).

```java
public static Calendar getLongFromDateTime (String date, String dateFormat, String time, String timeFormat) {
  Calendar cal = Calendar.getInstance();
  Calendar cDate = Calendar.getInstance();
  Calendar cTime = Calendar.getInstance();
  SimpleDateFormat sdfDate = new SimpleDateFormat(dateFormat);
  SimpleDateFormat sdfTime = new SimpleDateFormat(timeFormat);
  try {
    cDate.setTime(sdfDate.parse(date));
    cTime.setTime(sdfTime.parse(time));
  } catch (ParseException e) {
    LogDelegate.e("Date or time parsing error: " + e.getMessage());
  }
  cal.set(Calendar.YEAR, cDate.get(Calendar.YEAR));
  cal.set(Calendar.MONTH, cDate.get(Calendar.MONTH));
  cal.set(Calendar.DAY_OF_MONTH, cDate.get(Calendar.DAY_OF_MONTH));
  cal.set(Calendar.HOUR_OF_DAY, cTime.get(Calendar.HOUR_OF_DAY));
  cal.set(Calendar.MINUTE, cTime.get(Calendar.MINUTE));
  cal.set(Calendar.SECOND, 0);
  return cal;
}
```

Fig. 4 Source code of getLongFromDateTime()

```java
@Test
public void getLongFromDateTimeTest() {
  String date="2020-03-12";
  String dateformat="yyyy-MM-dd";
  String time="15:15:15";
  String timeformat="HH:mm:ss";
  Calendar calendar=DateUtils.getLongFromDateTime(date, dateformat, time, timeformat);
  assertEquals( expected: 2020, calendar.get(Calendar.YEAR));
  assertEquals( expected: 2, calendar.get(Calendar.MONTH));
  assertEquals( expected: 12, calendar.get(Calendar.DATE));
  assertEquals( expected: 15, calendar.get(Calendar.HOUR_OF_DAY));
  assertEquals( expected: 15, calendar.get(Calendar.MINUTE));
  assertEquals( expected: 0, calendar.get(Calendar.SECOND));
}
```

Fig. 5 Source code of getLongFromDateTimeTest()

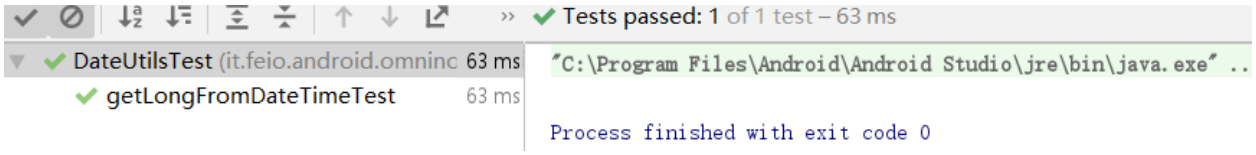

Fig. 6 Result of running getLongFromDateTimeTest()

2.3 TestTagWithComma()

In Omni-Notes, users can add customized tags to the notes. And this test case was built to test a special case of tags, tags with a comma. The source code of this test case is shown in Fig. 7. In this test case, we used addTagToNote() to add a tag with a comma to an existing note. After that, we used retrieveTags() to see whether we can separate the tag from the note correctly. Because there is a bug in this app and the system cannot identify tags with a comma correctly, we use an assertFalse statement to confirm the system cannot identify tags with a comma so far. The testing result is shown in Fig. 8. The test case passed, which indicates that the newly created tag with a comma indeed was not identified and added into the tag list of the system.

```
@Test
public void TestTagWithComma() {
    String newTag="#comma,comma";
    List<Tag> tags = new ArrayList<>();
    tags.add(new Tag(newTag, count: 1));
    Pair<String, List<Tag>> newTags = TagsHelper.addTagToNote(tags, new Integer[]{0, 1}, note);
    HashMap<String, Integer> tags1 = TagsHelper.retrieveTags(note);
    assertTrue(newTags.first.contains(newTag));
    assertFalse(tags1.containsKey(newTag));
}
```

Fig.7 Source code of TestTagWithComma()

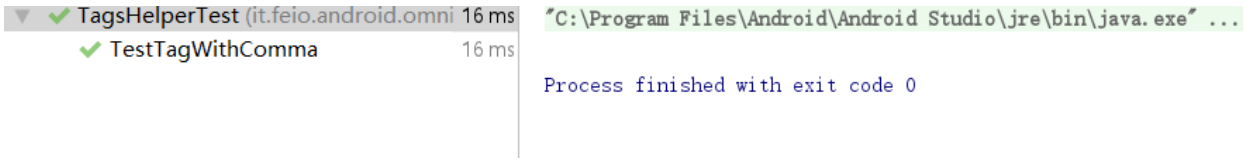

Fig.8 running result of TestTagWithComma

3 Pull Request

After we created the three test cases, we made a pull request for them (Fig. 9).

Fig.9 Pull Request of adding the new test case for testing tags with commas

4 New Understanding of the App

We developed three test cases for the existing methods in Omni-Notes. The first two test cases covered two methods - getDataFromString() and getLongFromDateTime() - that have not been tested yet. They are used to set reminders' time and add date and time to the notes, which is an important utility in the app. The getLongFromDateTime() must set both Date and Time, and it will automatically set the SECOND to 0. We believe the developer wanted the method to ignore the given SECOND and just set the date, hour and minute. So when the reminder or other functions use this method, they will always start at the same second. In comparison, the getDataFromString() can be used to convert either only Date or both Date and Time. Therefore, compared with getLongFromDateTime(), it is more flexible.

As for the third test case, we developed the test case when we tried to fix an open issue of the project published on Github. The problem is: tags with a comma cannot be identified. First, we tried to reproduce the issue. And after we added some tags with commas, we indeed cannot find the tags in the tag list. We found when retrieving tags from notes, the regex did not include commas. We believe this is the bug, and we decided to fix it. But after further reading the code, we realized the regex formula was written in a jar file. Because of this, we are not able to fix it. So we developed this test case. We thought after the author fixes this bug, he can easily use this test case to check whether the bug gets fixed.