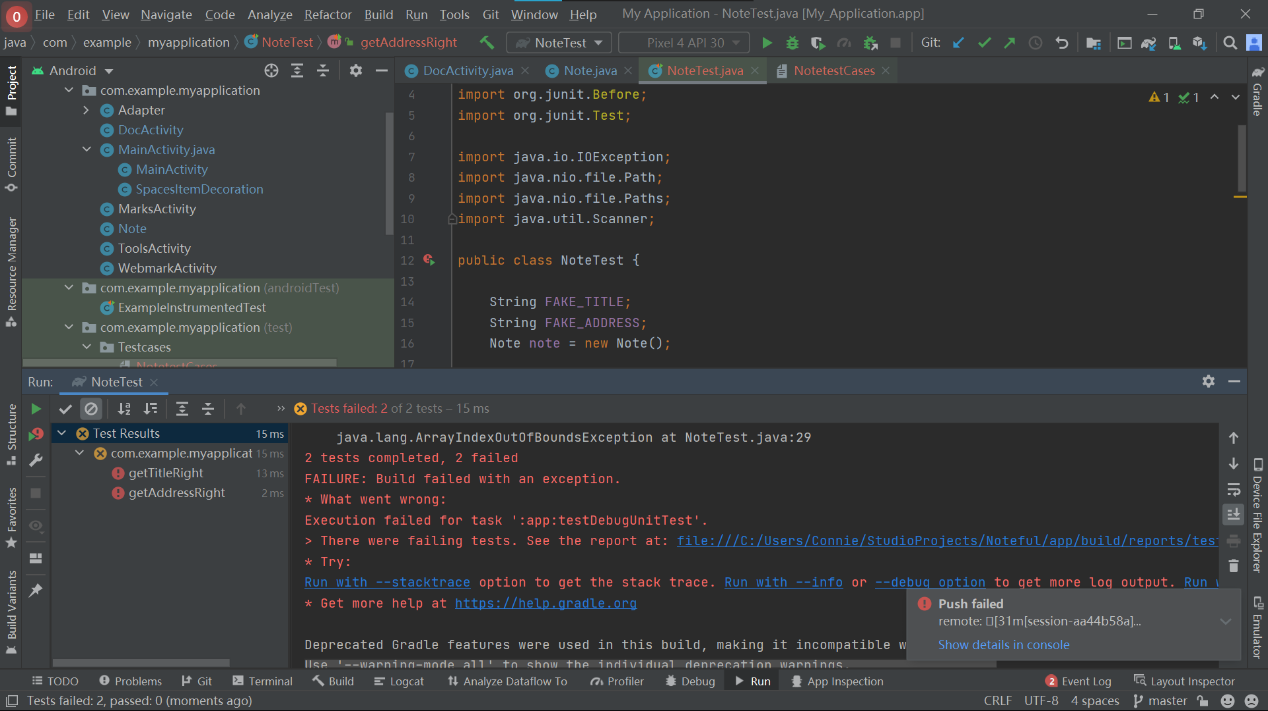
We have designed a unit test to test the correctness of the class Note in order to verify whether it has got the right Title and Address. With a void string, it failed to pass the unit test. And input a string with content, it passed the unit test.

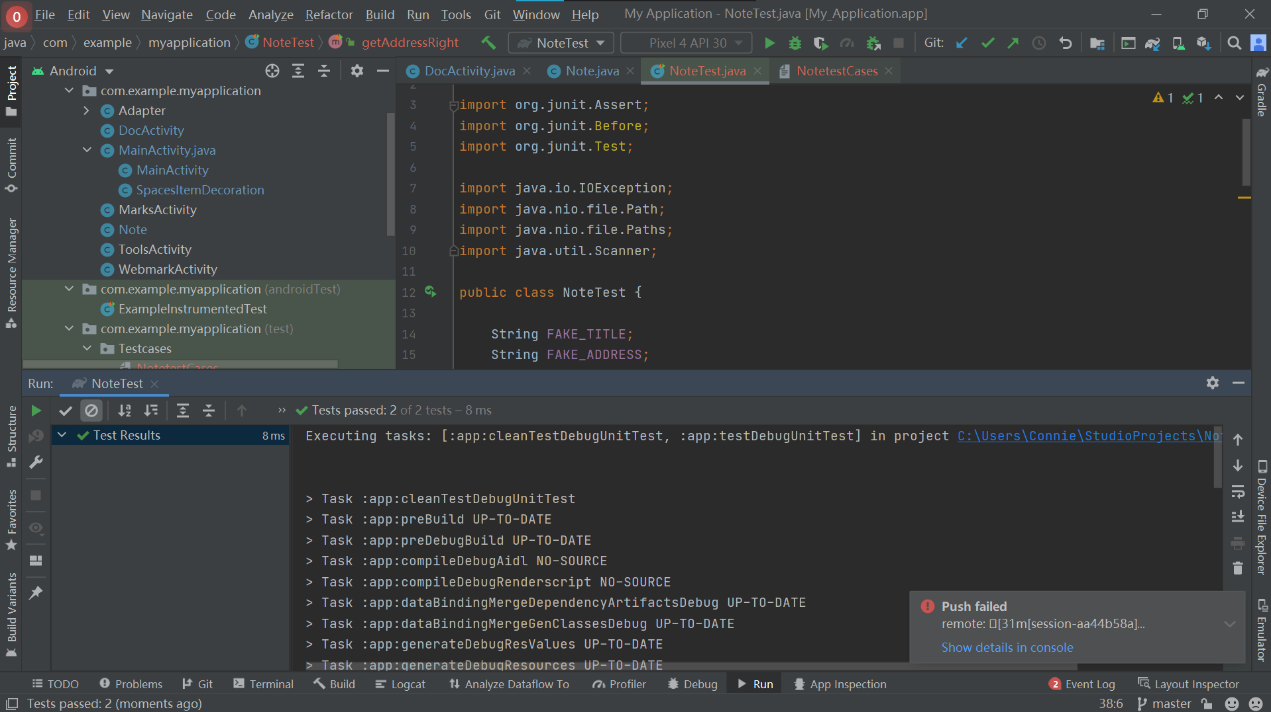
**Test cases**

|  |  |
| --- | --- |
| Title(content: String) | Address(content: String) |
|  |  |
| 0 | 0 |
| fdnsakf | 1 |
| vcnzk | 2 |
| fjsiow | 3 |
| fkoajv | 10 |
| ncsvnl | 5 |
| djfi | 12 |

Input: void



Input:string



**Code:**

package com.example.myapplication;

import org.junit.Assert;

import org.junit.Before;

import org.junit.Test;

import java.io.IOException;

import java.nio.file.Path;

import java.nio.file.Paths;

import java.util.Scanner;

public class NoteTest {

String FAKE\_TITLE;

String FAKE\_ADDRESS;

Note note = new Note();

public NoteTest() throws IOException {

}

@Before

public void setUp() throws Exception {

String fileName = "src/test/java/com/example/myapplication/Testcases/NotetestCases";

Path path = Paths.get(fileName);

Scanner scanner = new Scanner(path);

if((scanner.hasNextLine())){

String line = scanner.nextLine();

String[] line\_split = line.split("\\s+");

FAKE\_TITLE = line\_split[0];

FAKE\_ADDRESS = line\_split[1];

}

note.setAddress(FAKE\_ADDRESS);

note.setTitle(FAKE\_TITLE);

}

@Test

public void getAddressRight(){

String TEST\_ADDRESS = note.getAddress();

Assert.assertEquals(FAKE\_ADDRESS,TEST\_ADDRESS);

}

@Test

public void getTitleRight(){

String TEST\_TITLE = note.getTitle();

Assert.assertEquals(FAKE\_TITLE,TEST\_TITLE);

}

}