Unit test for Android:Flutter

March 2020

1 Introduction

This unit test document applies to the Integration, Converter and Notification sub teams of Team Gamma. For the purposes of performing unit tests for the android component(flutter/dart) we shall be making use of the built in unit test features provided by flutter.
The steps below describe an example of how you can create unit tests in Android studio/ IntelliJ.

2 Create a test file and importing flutter_test dependencies

In Android Studio under the package tab you will find a folder called 'test'. Within this folder there will already exist a template widget test file which you may have a look at and run it to see how it works.

Create a new dart file in this test folder and name it descriptively relating it to the specific function you would like to test for example if you would like to test a function that does something with an email address entered by a user then to test if the email address entered entered is valid or invalid you can name this test file "emailAddressValidator_test". Note that it is important to make sure the name of the file ends with **_test.dart** as that is the convention and format the test runner uses when searching for tests.

Once you have created this file open it and at the top of the file include this line of code:

import 'package:flutter_test/flutter_test.dart';

as well as import the file where the actual code of the function you are testing is located, for example if the above enter email address function is located in main.dart then you will add this line at the top of your testfile as well where

project_name is the name of the project you entered when creating a new project:
import 'package: [project_name]/main.dart';

3 Writing a test file

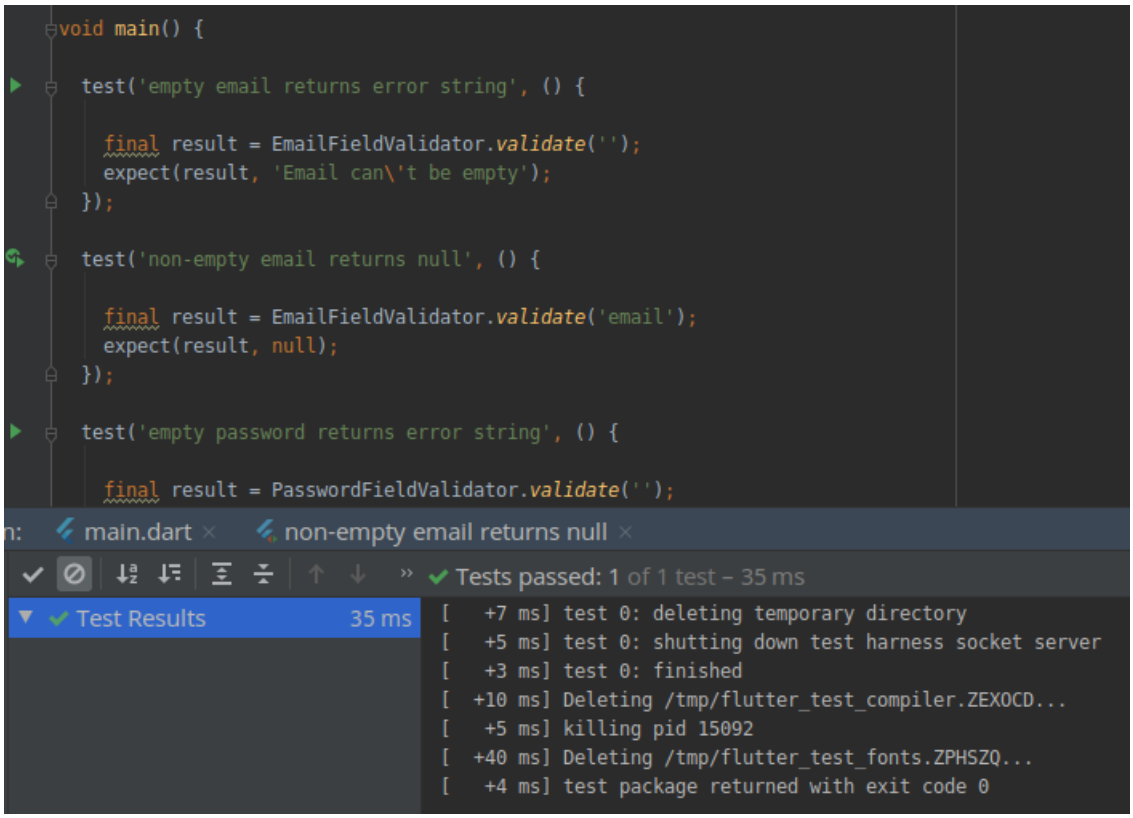
Tests are defined using the top-level 'test' function, and you can check if the results are correct by using the top-level 'expect' function. For example to test that the email address entered by the user is not empty you can define the tests in this manner.

```dart
import 'package:flutterapp/main.dart';
import 'package:flutter_test/flutter_test.dart';

void main(){

  test('empty email returns error string', () {

    final result = EmailFieldValidator.validate('');
    expect(result, 'Email can\'t be empty');
  });

  test('non-empty email returns null', () {

    final result = EmailFieldValidator.validate('email');
    expect(result, null);
  });
```

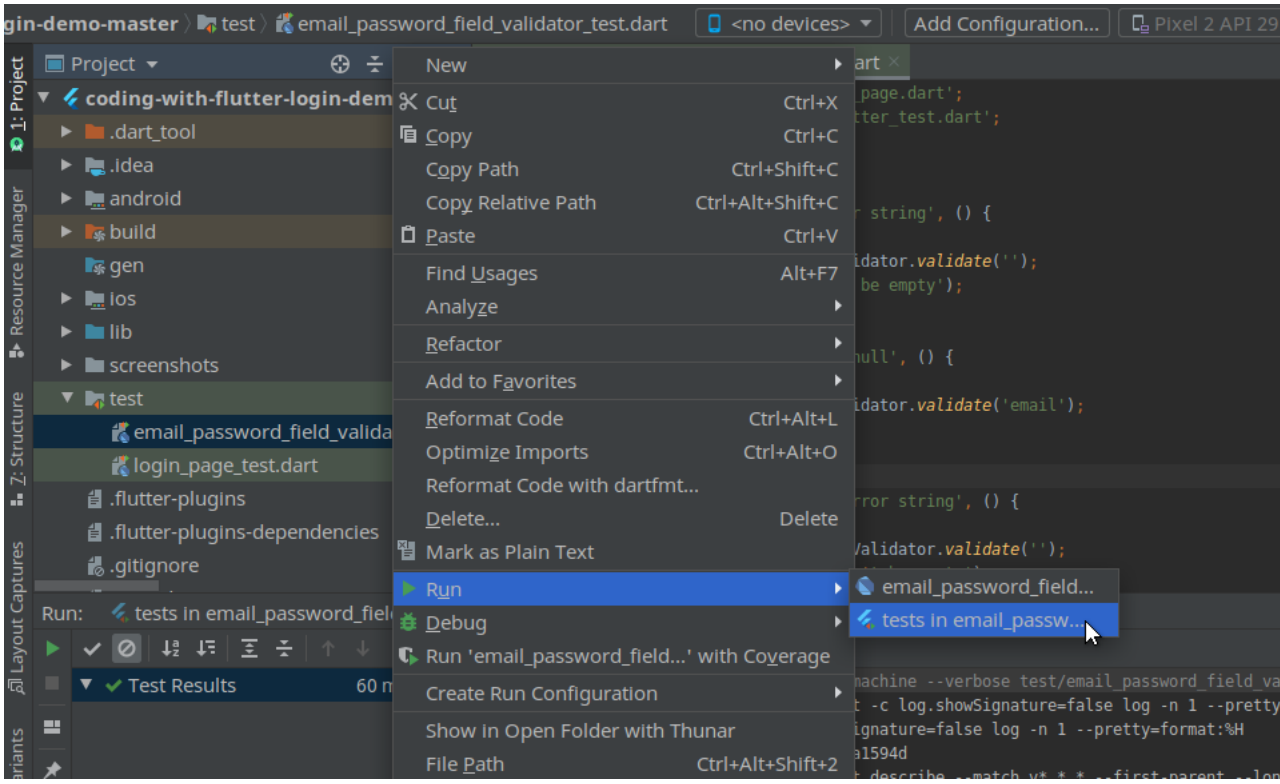
The **'test'** function takes 2 arguments, a description of what the test is about and a function that is defined which is where the actual test code is entered. The **'expect'** function takes 2 arguments, the first is the value returned by the function being tested above and the second is a predefined output which it can be tested against.

4 Running the test file

In Android studio you will also find green play/run buttons on the side next to each test function as seen below which can be used to run only that specific test if required and if in VS Code you can make use of the debug feature to run the test file.



If you would like to run all of the tests in a file you can do so using this method shown in the image below by right clicking the test file, select run and then select tests in [filename] or another method you can use is by grouping certain test functions together as explained in reference 2 in the reference section below.



5 References

- 1. https://www.youtube.com/watch?v=h0IbAIHAwnk
- 2. https://flutter.dev/docs/cookbook/testing/unit/introduction
- 3. https://flutter.dev/docs/testing