# Learning Model for Android Programming Learning Assistant System

Yan Watequlis Syaifudin[1] Nobuo Funabiki[2] Minoru Kuribayashi[3]

p52m5jeb@s.okayama-u.ac.jp [1] funabiki@okayama-u.ac.jp [2] kminoru@okayama-u.ac.jp [3]

Graduate School of Natural Science and Technology, Okayama University

**Abstract** With increasing demands for *Java-based Android application engineers*, we have developed the *Android Programming Learning Assistant System (APLAS)*. In this study, we propose a learning model that can guide students to independent learning by utilizing the *test-driven development (TDD)* method to automatically mark the student answers, where *JUnit* and *Robolectric* are adopted as the marking engines. The application to 40 students proves the effectiveness of the proposal in learning Android programming.

## 1 Introduction

With the increasing use of Android devices, the need for Android mobile applications is increasing. As a result, the demand for Android developers has become one of the highest ones in the IT field. To improve Java-based Android programming learning environments, we have developed the *Android Programming Learning Assistant System (APLAS)*. Using the *Test-Driven Development (TDD)* method *TDD*, APLAS offers the automatic marking of student answers that can guide students to independent learning and reduce burdens of teachers. In APLAS, a student will be given a set of tasks to make an Android project referring to the specifications given in the task guide written in natural languages. Then, the results of the tasks or answers are verified by the corresponding test codes built by the teacher on *JUnit* [2] and *Robolectric* [3] software APIs.

In this paper, we propose a learning model for APLAS. In *Android programming learning*, a lot of topics must be learnt. Here, we develop the step for learning the basic topics of the user interface. The user interface is very important for realizing interactions with a user. To evaluate the proposal, we applied the system to 40 undergraduate students in Indonesia.

## 2 Related Works

In 2011, Sadeh et al. conducted the evaluation of unit testing for android applications using Robolectric [4]. Their study showed that *Robolectric* is useful in conventional instrumentation testing.

In 2013, Funabiki et al proposed a Web-based *Java Programming Learning Assistant System (JPLAS)* for learning Java programming using the TDD method [5] using *JUnit*. It can enhance educational effects in Java programming courses by allowing self-studies of students while reducing teacher loads.

In 2015, Kang et al conducted the research on Android programming education using *Multi Android Development* Tools [6] that use *MIT App Inventor* and *Eclipse*. For beginner students, it is easy to use puzzle models provided in *App Inventor* but the teacher must manually check the code validity.

## 3 Test-driven Development Method

In this section we will review the *Test-Driven Development Method (TDD)* that utilized in *APLAS*.

### 3.1 TDD Method

The TDD method is a software development process that relies on repetitions of a very short development cycle [1]. Requirements are turned into very specific test cases. Then, the program code is improved to pass the new tests. In the Android application, there are three types of tests, namely the unit test, the integration test, and the *UI test*. In this paper, we will focus on the unit test, because it can test particular parts of the application object.

### 3.2 Unit Testing

The unit testing is a practice to test small, individual, and isolated units of a code [1]. It tests a small unit in an application such as a method and a class in Java programming case. The test is written by some programmers for other programmers [7]. The unit testing is useful to prove the validity of the code unit that is a small part of the entire application code.

### 3.3 Android Application Testing Framework

*JUnit* is the de facto standard for unit tests on Android [7], and is the main framework easy to use for the automated unit testing. For the *Android UI test*, *Robolectric* provides the framework that brings unit tests. The *Android UI test* runs on *Java Virtual Machine*, such that the process runs faster than on the emulator.

## 4 Learning Model

As the initial stage of developing *APLAS*, this study focuses on the *Basic User Interface* as the most basic level of Android programming learning.

### 4.1 Learning Process

Here, a student learns how to make UI for an Android application by solving the given questions on UI

specifications using *Android Studio*. Then, with the test code given from the teacher, a student runs the answer code to mark it. Fig. 1 shows the process.

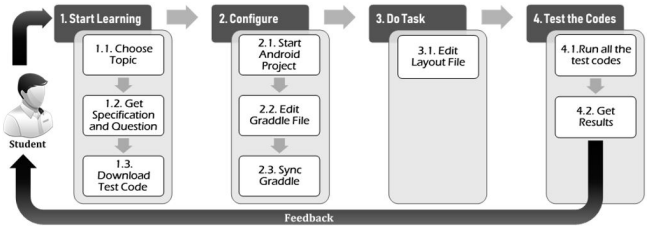


Fig. 1 Basic UI learning process.

### 4.2 Unit Testing

Referring to Horton's book [8] and Udacity online learning site [9], there are three aspects that must be understood to start an Android application development project, including 1) Start Android project (package, *Android SDK*, application compatibility, and *graddle* knowledge), 2) Configure the resources (Android project resource types), 3) Design the UI (*layout building* paradigm).

There will be nine tasks in Table 1 that must be completed sequentially to build an application display including all of the three aspects. When a student completes the entire task, *a unit converter application* display will be generated.

Table 1 Learning topics.

| Task No. | Title | Aspect |
|---|---|---|
| 01.01 | Project configuration | Start project |
| 01.02 | Resource configuration | Resource |
| 02.01 | *Main layout*, *textview*, *button* | UI |
| 03.01 | *Space* and *Child layout* | UI |
| 03.02 | *String-array*, *EditText*, *Spinner* | Resource, UI |
| 04.01 | *Checkbox* | UI |
| 04.02 | *RadioGroup* | UI |
| 05.01 | *Image resource* and *ImageView* | Resource, UI |
| 06.01 | *Drawable resource* and *Table layout* | Resource, UI |

### 4.3 Validation Process

A student who has completed the task must validate it using the unit testing method in Fig. 2:

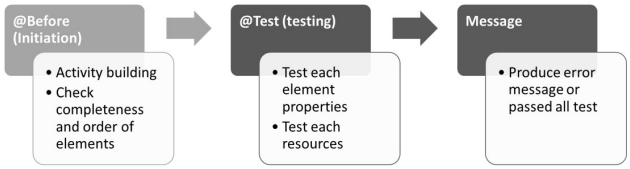


Fig. 2 Validation process.

The process including: 1) Check the completeness of all the layout elements and resources, then check the order of the placements. 2) Test all the values of each property in all the resources and layout elements. 3) If there is a failed test, a message will appear.

## 5 Evaluation

To evaluate the proposal, we asked 40 IT students to solve the assignment tasks using Android Studio during a class. We recorded the completion time and the feedback of each task. Table 2 shows that the majority of students completed the tasks and said easy to do. This proves that our model is effective to use.

Table 2 Implementation Result.

| Aspect | Result |
|---|---|
| Completion | 87% Passed, 13% Failed |
| All Duration | Fastest: 81 minutes, Longest: 385 minutes, Average : 216.0 minutes |
| Difficulties | Easy: 03.01, 04.01, 04.02, 05.01<br>Normal: 03.02<br>Hard: 01.01, 01.02, 02.01<br>Very Hard: 06.01 |
| Most comment | Easy to do |
| Most difficult | Hardware and internet connection |
| Failed Problem | Internet problem,<br>Student was not familiar with XML code,<br>Last task was most difficult task. |

## 6 Conclusion

This paper presented a learning model utilizing the TDD method to automatically mark the student answers. The experiments confirm the effectiveness. In future, we will implement the database and *service interactions*.

## References


[1] V. Farcic and A. Garcia, Test-Driven Java development, Packt Publishing, 2015.

[2] https://junit.org/.

[3] http://robolectric.org/.

[4] B. Sadeh and S. Gopalakrishnan, "A study on the evaluation of unit testing for Android systems," Int. J. New Comput. Arch. Appli, 2011

[5] N. Funabiki, Y. Matsushima, T. Nakanishi, K. Watanabe and N. Amano, "A Java programming learning assistant system using test-driven development Method", IAENG Int. J. Comput. Sci., vol. 40, no. 1, pp. 38-46, 2013.

[6] H. Kang and J. Cho, "Case study on efficient Android programming education using multi Android development tools", Indian J. Sci. Tech., vol. 8, no. 19, pp. 1-5, 2015.

[7] P. Blundell and D. T. Milano, Learning Android application testing, Packt Publishing, 2015.

[8] J. Horton, Android Programming for beginners, Packt Publishing, 2015.

[9] https://www.udacity.com/course/android-basics-nanodegree-by-google–nd803.