Indian Journal of Science and Technology, *Vol 8(19), DOI: 10.17485/ijst/2015/v8i19/75984, August 2015*
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Case Study on Efficient Android Programming Education using Multi Android Development Tools

Hwansoo Kang* and Jinhyung Cho

School of Computer Engineering, Dongyang Mirae University, Seoul-152-714, South Korea;
hskang,cjh@dongyang.ac.kr

Abstract

The purpose of this study is to propose education model using Eclipse ADT with android SDK and App Inventor together for efficient teaching for developing Android application. Android is an operating system for mobile devices, one of the most used in the world today. Android is now used in more devices, including smart phones and more users than any other mobile operating system. Android SDK is required to Android app developer, but students are difficult for Android SDK because Android SDK requires students the prior knowledge of Java programming language and Android SDK class library. MIT App Inventor is a web based tool, which allows users with no programming experience to develop mobile apps. The Android programming education model designed in this study was applied to a course for mobile apps. This model using multi Android development tools have shown to be more efficient compared to education using Android SDK only.

Keywords: Android Programming Education, Android SDK, App Development Tool, App Inventor

1. Introduction

As operating system of the Android is getting more and popular, people are getting more interested in applications based on the Android SDK. Now the Android system in the smart phone market is becoming more and more popular especially. In recent years, there are many applications generated because some of the development tools are free owing to the open source[1]. In general, in order to develop Android apps, there are a few prerequisites that you need to fulfill. Those are Java and XML experience, and development environment such as Eclipse. Therefore students are difficult for learning Android apps. But MIT App Inventor, which is an Android app creating tool, provided by MIT that turns the app development process into a visual one. App Inventor is great for non-coders and beginners. However, in order to become a professional app developer, it is necessary to develop an app using Android SDK.

2. Android App Development Tools

2.1 Android SDK

Android is a open source mobile operating system based on Linux kernel developed by Google of OHA_(Open Handset Alliance) to develop Android mobile apps[2]. To develop Android mobile apps, a set of tools that are included in the Android SDK are needed. There are the API libraries and developer tools necessary to build, test, and debug apps for Android in the Android SDK. Generally we use Eclipse IDE with Android SDK, and related tools for developing Android app. The ADT Bundle provides everything you need to begin developing apps. The essential Android SDK components and a version of the Eclipse IDE with built-in ADT (Android Developer Tools) are included in the ADT Bundle to streamline your Android app development[3]. The ADT Bundle shown in Figure 1 is widely used to develop Android mobile apps.

*Author for correspondence

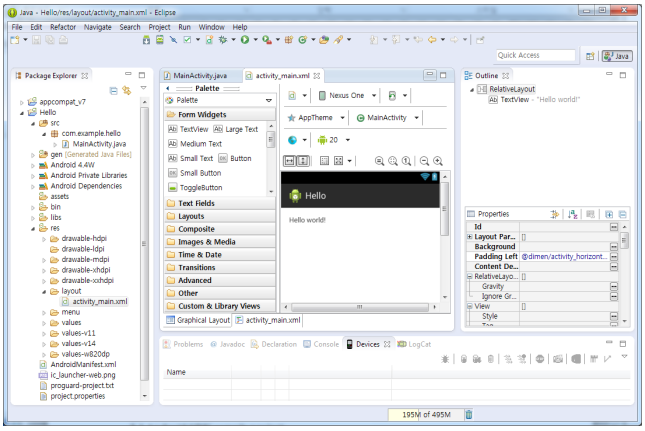


Figure 1. Screen of ADT Bundle development environment.

The basic steps for developing applications are four steps. Developing applications includes the following four steps[4,5]:

- Setup: During Setup phase we install and set up our development environment. We also create AVDs (Android Virtual Devices) and connect hardware devices, on which we can install our applications.
- Development: During Development phase we prepare and develop our Android project, which contains all of the source code ands resource files for our application.
- Testing and Verifying: During this phase we build our project into an .apk package, which is installed and running on the emulator for debugging.
- Publishing: During this phase we prepare and build our application for release and distribute release version of our application to users.

2.2 MIT App Inventor

MIT App Inventor is a web-based, a drag-drop visual development environment where people can quickly create Android mobile apps by plugging together program blocks with a graphical user interface[6,7]. When App Inventor was announced in July 2009 as an experimental teaching and learning tool, the developer tool was mainly aimed at giving pupils and students easy access to programming in general and mobile devices in particular. App Inventor that is a visual 'blocks' programming language for creating mobile apps, is an developing environment for developers lacking in programming experience[8,9]. Its graphical blocks programming is based

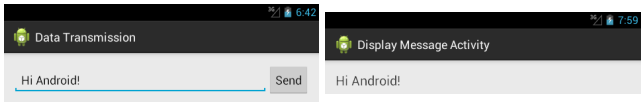
Figure 2. Screen designs of two activities.

on the Massachusetts Institute of Technology's Scratch programming environment. App Inventor is a visual tool consisted of the Component Designer and the Blocks Editor shown in Figure 2[10]. The Component Designer of App Inventor designs the App's user interface by arranging both on and off-screen components. The Blocks Editor of App Inventor programs the app's behavior by putting blocks together[11,12].

3. Android Programming Education using Android SDK and App Inventor

3.1 Android SDK Sample Project

We show one sample project for efficient Android programming education using Android SDK and App Inventor. This project is sending string data to another activity. Figure 3 is screenshots of the sample project.



(a) **(b)**

Figure 3. Screen designs of two activities **(a)** Screen design of main activity and **(b)** Screen design of another activity.

Skills needed for this project are intent and event handling. When a user interacts with components, events are occurred. Event handling is a useful procedure to process the event. We use the layout xml file to specify the event handler method send Message via the android: on Click attribute for click event and implement event method send Message () in activity source file. Figure 4 is the content of the UI xml and activity source file. Intent is responsible for communications between two activities in the Android system. Among four components, which are activity, broadcast receiver, service and content provider, messages and data are transmitted via intent.

We can receive the Intent of our activity by calling get Intent() and access the data belonged to it. In another activity, classes on Create () method of Figure 5, get the intent and extract the message delivered by main activity by calling get String Extra().

3.2 App Inventor Sample Project

Development app to the above functions using App Inventor makes it easier to understand skills effectively. The basic elements for creating Android apps are components in the App Inventor. We need to click a component and drag it onto the viewer in appropriate position in the Designer for organizing a component in our Android app. The project is composed of two screens. Figure 6 show the screens of App Inventor for this project.

The first screen 'Screen1' is composed of a Text Box and Button, the second screen 'Screen2' is only composed of a Label. On the User Interface palette, we drag and drop the Text Box and Button component to the Viewer, placing the min 'Send a Message to Another Activity' screen to configure the Screen1. Likewise we drag and drop the Label component to the Viewer, placing it in 'Display a Message' screen to configure the Screen2. So far we have been disposing our app's screens and components in the Designer, which is running in a web browser. To start the behavior of the app programming, we need to edit in the Blocks Editor. On the left side of the Blocks Editor, we click the Button1 of user interface to open it. We drag the Button1.Click block in the work area and drop it on the open area on the right. The event handler of blocks describes how the phone should respond to specific events. For example, if Button1 is pressed, 'when Button1.Click' is an event handler. To implement the event handler, appropriate blocks must be inserted into the block of 'When Button1.Click'. Control Block of 'open another screen with start value screens Name and start Value' is inserted into the block of 'When Button1.Click'. Also we insert 'Screen2' Text Block into screen Name, and 'TextBox1.Text' Block of Component TextBox1 into start Value. The yellow block is called

UI XML file

```
<Button
    android:id="@+id/buttonSend"
    android:onClick="sendMessage"
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Send" />
```

Send

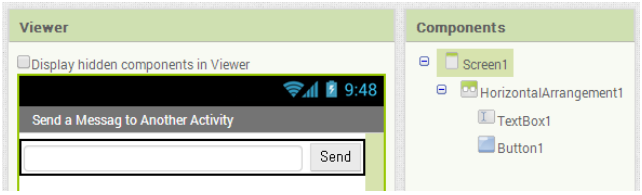
Source file

```
/** Called when the user clicks the Send button */
public void sendMessage(View view) {
    // Do something in response to button
    Intent intent = new Intent(this, DisplayMessageActivity.class);
    EditText editText = (EditText) findViewById(R.id.editMessage);
    String message = editText.getText().toString();

    intent.putExtra(EXTRA_MESSAGE, message);
    startActivity(intent);
}
```

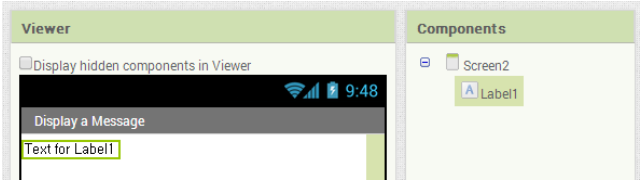
Figure 4. Source code for start an activity.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.display_message);

    // Get the message from the intent
    Intent intent = getIntent();
    String message =
            intent.getStringExtra(MainActivity.EXTRA_MESSAGE);

    // Create the text view
    TextView textView =
            (TextView) findViewById(R.id.displayMessage);
    textView.setText(message);
}
```

Figure 5. Source code for receiving data.



(a)



(b)

Figure 6. Screen designs of two activities **(a)** Screen1: Send a Message to Another Activity and **(b)** Screen2: Display a Message.

a control block, which is placed in the body of event handlers of 'When Button1.Click'. As Blocks shown in the screen1, to set text with a start value into Label 1 in the screen2, a Block of 'set Lable1.Text' set 'get start value' inserted into the block of 'When Screen2. Initialize'. Figure 7 shows blocks program for button click event and displaying the received data.

4. Application of the Model to a Course

The Android programming education model designed in this study was applied to a course opened in the second semester of 2013. The title of this course was 'Mobile Programming' for android programming. This course had 3 hours per week and was focusing on the practical android programming exercises using in parallel with traditional Android SDK of Eclipse ADT (Android Development Tool) Bundle and MIT App Inventor. Table 1 describes the proposed course syllabus for Android app programming. The first week and second week of this course, we introduced the syllabus and Android OS, Android SDK. At the third week, we introduced App Inventor and learned app programming with App Inventor**.** In 4-13 weeks, we were learning in app programming using App Inventor and Android SDK in parallel. At the end the 14th week of the course, term project presentation was held for Capstone Design team projects of Android app planned, designed and developed by students themselves. At first, this project was developed using App Inventor and then must be re-built using the Android SDK. According to the course evaluation result, this model using multi android development tools has shown to be more efficient compared to education using Android SDK only.

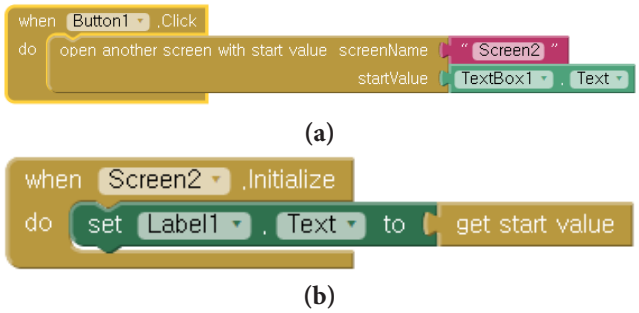


Figure 7. Blocks for two activities **(a)** Blocks programming for start an activity and **(b)** Blocks programming for receiving data.

Table 1. Course syllabus

Week	Android Projects	Tools
01	✧ Overview of the Course ✧ Overview of the Android OS and Android programming	*Android SDK*
02	✧ Installation Development Environments, Android SDK ✧ Hello World programming exercise	*Android SDK*
03	✧ Overview of the App Inventor ✧ Overview of Designer and Blocks Editor of App Inventor ✧ Hello World and Hello Kitty programming exercises	*App Inventor*
04	✧ Activity, View and Layout ✧ List View, Spinner and Date Picker	*Android SDK*
06 ~ 07	✧ Event Handling of SDK ✧ Event Handling of Blocks programming of App Inventor	*Android SDK App Inventor*
08	Mid Test	
09 ~ 10	✧ Activity Lifecycle ✧ Data Transmission	*Android SDK App Inventor*
11 ~ 13	✧ Internet, Socket and Parsing ✧ SQ Lite Database Processing	*Android SDK App Inventor*
14	✧ Final Presentations	*Android SDK App Inventor*
15	Final Test	

5. Conclusion

Generally we develop Android app using Eclipse IDE with Android SDK. But it is difficult for beginner students to learn app programming. We like developing Android apps with Eclipse better because the developing tools, which you need while developing applications, can be directly invoked by it. Also we need Android SDK in order to be an expert in app programming. Therefore teaching app programming using Android SDK is indispensable for students. Finally we select the App Inventor to complement to teach app programming using Android SDK. We show that blocks programming edits will be more manageable and informative than the low-level nature of keystroke edits used in learning for Android app programming. Education of Android app development is easy for beginners of Android app development, if the

conceptual understanding for Android app development is preceded in education of Android app development. However it is difficult for beginners of Android to learn various development concepts for Android app development directly, in Android app education of using only the Android SDK. It is very effective for App development education to learn simple objects, components, and event handling, etc while developing App simply using App Inventor before developing App using Android SDK. It is more effective for beginner students to learn Android app development using the Android SDK after developing Android app using App Inventor than education using only the Android SDK. In addition, this efficient Android programming education is checked by the qualitative interviews of students in lecture of Android app programming course in 2013. The Android programming education model designed in this study, using multi Android development tools have shown to be more efficient compared to education using Android SDK only.

6. Acknowledgement

This study has been supported by 2014 Academic Research Project funded by Dongyang Mirae University in the Republic of Korea.

7. References

1. Ma L, Gu L, Wang J. Research and development of mobile application for android platform. International Journal of Multimedia and Ubiquitous Engineering. 2014; 9(4):187–98.
2. Pandey G, Dani D. Android Mobile Application Build on Eclipse. International Journal of Scientific and Research Publications. 2014; 4(2):1–2.
3. Android Developers Home Page, Android SDK develop tools download. 2014. Available from: http://developer.android.com/sdk/index.html
4. Android Developers Home Page, Android SDK develop tools workflow. 2014. Available from: http://developer.android.com/tools/workflow/index.html
5. Wasserman AI. Software engineering issues for mobile application development. Proceedings of the FSE/SDP workshop on Future of Software Engineering Research (FoSER '10); 2010. p. 397–400.
6. Kim B. Computer programming education using app inventor for android. Journal of the Korea Institute of Information and Communication Engineering. 2013; 17(2):467–72.
7. Cho E–S, Kim C–J, Lee S–H. A modeling technique for development of mobile app. based on android. Journal of the Korea Academia-Industrial cooperation Society. 2013: 14(8):3999–4005.
8. Park J. A study on the effect of app inventor in introductory android programming course. Proceedings of Korea Society of Computer Information Summer Conference 2013. 2013; 21(2):287–8.
9. Wolber D. App inventor and real-world motivation. 42nd ACM Technical Symposium on Computer Science Education (SIGCSE' 11); 2011. p. 601–6.
10. Kloss JH. Android apps with app inventor: The fast and easy way to build android apps. Addison-Wesley; 2012.
11. Getting Started with MIT App Inventor 2. 2013. Available from: http://appinventor.mit.edu/explore/get-started.html
12. Park J–Y, Park S–M. Android app development system using modular method. Journal of Korea Multimedia Society. 2014; 17(5):601–12.