International Journal of Multimedia and Ubiquitous Engineering
Vol. 9, No. 11 (2014), pp. 273-282
http://dx.doi.org/10.14257/ijmue.2014.9.11.27

Research and Implementation of Android Embedded Code Generation Method based on Rule Model

Wen Hu and Kai Zhang

School of Computer and Information Engineering, Harbin University of Commerce, 150028
wszk66@126.com

Abstract

Based on the in-depth study of the Android embedded application program frame and project file, this paper proposes a code generation method based on model rule, the method discusses the problem of the layout of the Android embedded application interface, defines the constraint rules between the components and component object model and event model. And code generation rules on the Android embedded applications are defined and used throughout the code generation process, it solves the visualization development of Android embedded application program and the code automatic generation. Finally, a simple examples is given to verify the process of automatic generation of Android embedded code.

Keywords: *Rule model; Code generation; Android*

1. Introduction

Android operating system becomes so popular and more and more developers dedicate to android app application development [1]. In the process of the development of mobile terminal program, many interface developments of mobile terminal program can be designed by development tools. Such as Microsoft Visual Studio 2005(VS2005), it has the function of visual development WinCE programming interface. Developers can design requirements of WinCE program interface and develop the frame structure by dragging and dropping the required relevant components and setting the attributes of the components. Developers can also use this interface development tools to write event code. For development of Android embedded applications, many developers use Eclipse development tools, although Eclipse development tool provides the function of the visual interface development, developers always write directly Android embedded XML interface code. Eclipse development does not have the function of generating the Android application event code and the Android structure [2].At present, some Android embedded development tools that do not provide the development methods for I \ O widgets in Android system and this part of the development is more complex and difficult [3, 4]. It takes developers a lot of time in developing programs and increases the development and maintenance costs. These development tools unrealized mainstream WYSIWYG mode of development.

2. Problem Analysis and Related Research

So much attention has been attracted by developer, which to develop a reliable Android application in a short time. The software development of visualization and code automatically generated become an important way to solve this problem [5].Interface design and the method of automatic generation model become a hot topic.

The user interface design that based on the model is the one aspect of attention [6], interface model is roughly divided into conceptual model and state model. Conceptual model is discussed from the angle of abstract logic relations between system structure and elements of the interface, it belongs to the theoretical significance, lacks of design oriented formalization, and depends on the cognition of the designer. The typical representatives are the Seeheim MVC, PAC and so on. Statement model uses information Element as analysis object and build a description of the interface through the interactive decomposition. Statement model is difficult to meet the needs of the complexity of the user interface and personalized, because of the lack of accurate description of the interface layout. Tadeus. Trident and Mastermind are the applications based on this model [7]. The main problem of the method based on model is the lack of the support from the description of abstract interface to implementation of specific interface, they can provide design description and recommendations but fail to provide concrete and effective support for the ultimate interface generation and layout.

The design of Android embedded application uses the MVC framework, that is to say, the user interface design and event code are separated, the UI (layout directory) generated by XML. Certainly, Android UI design of the embedded application can also be generated by writing java code, but the official advocate developers use xml file for view design. While application code is written in the Java file. The design of Android embedded application UI is composed of layouts, widgets and their mutual relations. Such as the widget dimension constraints and the hierarchical relationship between the widgets and layouts [8]. The constraint rules mapped to the XML code is the hierarchical structure of the UI code and attribute. It plays a guiding role for XML code of UI. In the process of dragging and dropping widgets or layouts to the canvas and the operation of adding Listener Events for widgets, user need to determine the type of widgets, then generate the corresponding Java code file. The relevant code can be accurately generated only by a series of rules to control. At the same time, Java files have the standard structure. The generated code block must follow certain rules and follow the Android embedded application file structure standard stitching, only in this way can the program compile.

In addition, the Android embedded application program has the high requirements of the user experience, there are many different forms of UI design, and developers to different design requirements can be accurately achieved. Based on these features, this paper proposes an Android embedded code automatically generated method that is based on the rule model. The method constraint the structure of the program and control the process of program generation by defining rules [9]. Under the guidance of the rule model, it achieves visual development of Android embedded and automatical generation for code. Workflow is shown as the Figure 1.

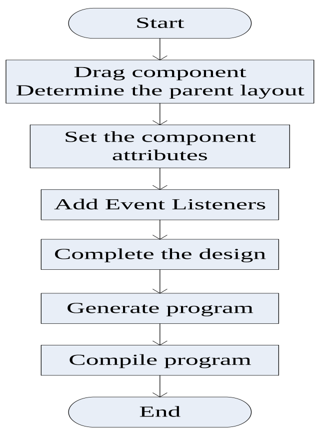


Figure 1. Development Workflow

3. Android Embedded Code Generation Based on Rule Model

3.1. Android Embedded Component Object Model

Android embedded component object refers to the carrier of operation flow and data flow when the program runs, it is also the objectification description of UI component. Android embedded components include layouts and widgets. The widget includes Android embedded commonly used widgets as Button, Text view and I/O widgets as LED Light, stepper motor, etc. Android component object model can be expressed as the following form:

Definition 1: Component Object Model, referred to as COM.

COM::=< ID, CO_ Name, Attribute _ set, SubCO_set, EventSource_Flag >

In the component object model：

(1) ID is the unique identifier for the component object. Such as TextView0l.

(2) CO_Name represents the type name of the component object.

(3) Attribute_set represents a set of attributes. Such as the location, size of Components. In Attribute_set, each attribute can be defined as the Attribute: = < key, value >. The key of the format is commonly" Android: text=", the value is a component object attribute values, can be a string.

(4) SubCO_set represents a collection of sub-objects that are contained in objects. Normally only the layout objects have child objects, but some widgets also have child objects, for example the RadioGroup. The widget of RadioButton is typically used in RadioGroup.

(5) EventSource_Flag is identified for event source. If EventSource_Flag = True that this component can be added to event listener as the event source. Otherwise, this component cannot be added to event listener.

3.2. Android Embedded Event Model

For the purposes of android event based on listener, the main approach is to bind the specific event listeners for Android UI components. In the processing model of event listeners mainly involves three kinds of objects: EventSource, Event, EventListener [10]. In the process of Android embedded event code automatically generated, in order to accurately generate an event code, according to the progress of processing Android event listeners to abstract listening events as event model, the event model expression is as follows:

Definition 2: EventModel referred to as EM. EM::=< EventSource, SetListener,Listener, Event_Content> .In the EventModel:

(1)EventSource is the place where events happen; it is usually the individual component, such as button.

(2)SetListener is the statement for component that need bind event listeners. Such as setOnClickListener().

(3)Listener represents an event listener and event handler framework.

(4)Event_Content represents event handling methods.

Android event model is an abstraction for Android event handling process.When generating event-handling code, according to the framework of the listener event model to make up events code block, consequently the event handling code that conform to the standard of Java code is generated

3.3. Android Embedded Interface Rule Model

The user interface model of Android embedded application is a model of the Android system component in the interface in the form of organization and the special requirements of the constraint. User interface model is an abstract description of Android external UI and focuses on the interface of macroscopic description and the composition of interface.As a constraint framework for the entire interface, Android interface consists of a series of layouts and widgets and constitute a Layout-Widget interface model. Layout in UI model is the parent class of all layouts, with the general attributes of the layout. Widget in UI model is the parent class of all widgets, with the general attributes of the widget.The specific constraint rules between layout and widgets are given as follows.

(1)The type constraint is used to indicate the quantitative relationship between widget and layout. Type constraints can be divided into two situations, the definition of rules are given as follows:

Rule1: Layout cannot accommodate all the widgets, but all the layouts must contain widgets that has been placed.

Rule2: Layout can accommodate multiple widgets, but any widgets must be contained only in a layout.

(2)Size constraints are constraints on component size.

Rule3: The size of the child components cannot exceed the size of the parent component.

(3)Schema constraint is used to describe combination relationship between components. Schema constraint can be subdivided into two categories. Inclusion relationships: it indicates the constraints between the whole and local and expresses the master-slave relationship between components, corresponding Rule4. Dependencies relationships: it indicates the presence of priorities between the layout or widget when drag to canvas, corresponding Rule 5. For example in the TableLayout layout, the number of columns is controlled by adding a TableRow. Only under the premise of the added TableLayout components can add TableRow to increase the number of columns of the table. TableRow cannot used alone.

Rule4: If marked as B $\sqsubset$A.It represents that LayoutA is the parent layout of WidgetB or LayoutB, WidgetB or LayoutB inclusion in the LayoutA.

Rule5: If marked as A→B. It represents that the placed sequence of WidgetA and WidgetB, WidgetB must be added after WidgetA.

According to the constraints of the rules, Layout-Widget interface model structure diagrams is shown as Figure 2.

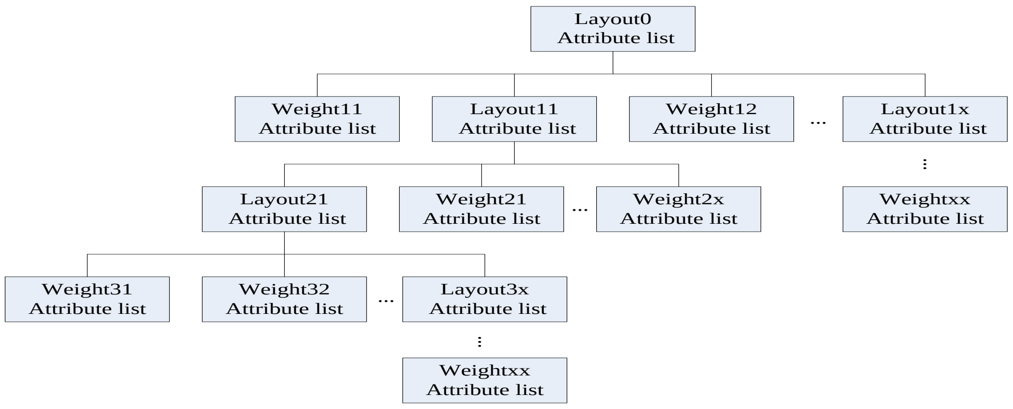


Figure 2. Layout-Widget Interface Model Structure

3.4. XML Generation Rules

Firstly, the users determine a root layout, then drag the layouts and widgets to the canvas and make sure the parent-child relationship between them. Users set and modify the attributes of the layout and widget objects, finally generated UI layout and Layout-Widget Interface model structure. Layout-Widget interface model is the abstract expression of the Layout interface application framework. XML code generation is guided according to the framework model, generating rules are given as follows:

Rule6: Writing XML file header information<?xml version="1.0" encoding="utf-8"?>then write the root layout's start tags and attributes to XML file. Next, the direct subcomponents in root layout are traversed. Jump to rule 7.

Rule7: If child components belong to the Layout, to write start tags and attributes of the component to XML file. if child components belong to the Widget, to write XML code according to the rules of 9. Each child component in parent component is traversed in turn.

Rule8: After all components which are in one layout are traversed each time, to write end tag to the XML file.

Rule9: If child component belongs to the widget, then write start tag, attributes and the end tag to the XML file.

After the above rules of constraints, the generation of XML code will be in the form of recursion, it greatly improves the efficiency of the XML code generation.

3.5. Java Files Generation Rules

The Android embedded Java file has a particular framework. The definition Java class format is divided into two parts: the definition of a class declaration and the class body. The statement includes the keyword class, the name of the class and class inheritance or other features. Class declaration format is as follows: [< modifier >] class < class name > [extends < parent name >] [implements < interface name >].Class subjects include properties and methods. The code of event listener is stitched according to event model framework.

The generated rules are definited for Android embedded Java files program as shown below.

Rule10: First of all, import package org.Droiddraw. The system determines the type of component, then adds the package name corresponding to the component to the package name collection according to the component type when the user drags the component to the canvas, while generating the declaration string of component object and add to the class attribute collection.

Rule11: Adding event listeners to components only when EventSource_Flag is true.

Rule12: Each event listener has a corresponding widget.

Rule13: The contents of event listener are generated based on EventModel of the component object and splice block according to its structure

Rule14: The variables of components are named in the form of string + number. Such as Button01 Textview02.This rule prevent a variable from defining repeatedly so that program cannot be compiled.

Rule15: The blocks of Java code that has generated are spliced according to the Java file structure , then generate the Java file that can be compiled.

4. The Experiment and Analysis

Experiment uses a simple Android application to illustrate the Android program code generation process. This instance implements a user who writes its name and clicks the Enter button to display the user name into the TextView. Using the visual code automatic generation tool make the widget and layout which is needed drag to the canvas and determine their level of relations between the components and set the attributes of layout and widget. Finally, program interface is formed as Figure 3.

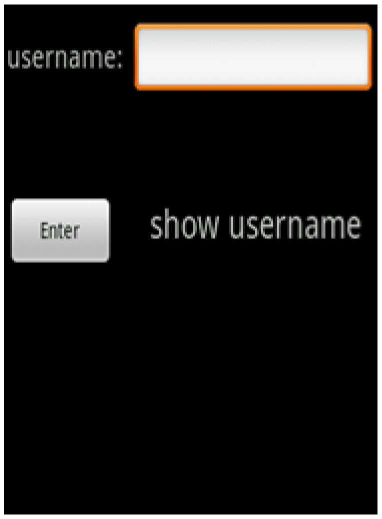


Figure 3. The Android Program Interface

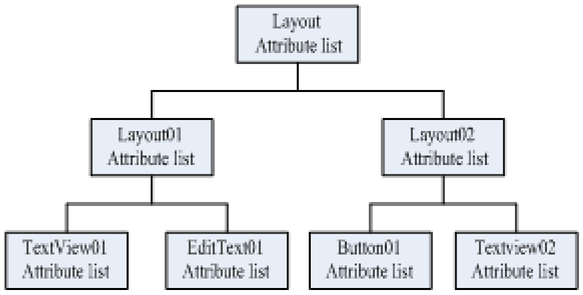


Figure 4. The Android Interface Model

4.1. XML Code Generation

UI components include a root layout which uses LinearLayout. There are two sub-layouts that are LinearLayout01 and LinearLayout02 in the root layout. The LinearLayout01 includes an EditView and a TextView called username. The LinearLayout02 includes a button named Enter and a TextView. Finally, generating the Layout-Widget interface structure (as Figure 4) is used to traverse XML interface and generate the code. XML code generation steps are given as follows:

Step1: According to Rule6: Write the XML file header information, the root layout's start tags and attributes to the XML file, the code is given as follows:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
   android:orientation="vertical"
   android:layout_width="fill_parent"
   android:layout_height="fill_parent">
```

Step2: Traversing the root layout's direct subcomponents, its subcomponents include two LinearLayout that are LinearLayout01,LinearLayout02.

Step3: According to Rule7: LinearLayout01,LinearLayout02 belong to layout. For each subcomponents in the operation, the beginning of the first is to write LinearLayout01's start tag and attribute to XML file. The code is given as follows:

```
<LinearLayout
   android:id="@+id/LinearLayout01"
   android:layout_width="wrap_content"
   android:layout_height="60px">
```

Step4: According to Rule7,continue to traverse LinearLayout01.There are two child widgets in LinearLayout01.They are TextView01 and EditText01.

Step5: According to Rule9,TextView01,EditText01 belong to widget. Writing TextView01 and EditText01's start tag and attribute to XML file, the code is given as follows:

```
<TextView android:text="username:   "
   android:id="@+id/TextView01"
```

```
  android:layout_width="wrap_content"
  android:layout_height="wrap_content"
  android:textSize="20px" >
</TextView>
<EditText
  android:id="@+id/EditText01"
  android:layout_width="200px"
  android:layout_height="wrap_content">
</EditText>
```

Step6:At this point, the widgets which are in the child layout named LinearLayout01 have been completely traversed. According to Rule8, writing LinearLayout01's end tag to XML file, the code is given as follows:

```
</LinearLayout>
```

Step7:UI code which includes LinearLayout01 and its child widgets have been generated. The LinearLayout02 which is in the same layer with LinearLayout01 start to be operated. There are two child widgets in LinearLayout02.They are Button01、TextView02. The operation of LinearLayout02 is like LinearLayout01. The code is given as follows:

```
<LinearLayout
    android:id="@+id/LinearLayout02"
    android:layout_width="wrap_content"
    android:layout_height="50px"
    android:layout_marginTop="50px" >
    <Button
        android:id="@+id/Button01"
        android:layout_width="90px"
        android:layout_height="wrap_content"
        android:text="Enter" >
    <Button/>
    <TextView
        android:id="@+id/TextView02"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="show username    "
        android:textSize="25px" >
     <TextView/>
</LinearLayout>
```

Step8:Until this step, the child layout which belongs to the root layout has been traversed. According to Rule8, writing the root layout end tag to XML file, the code is given as follows:

```
</LinearLayout>
```

Through the above steps, the example XML UI design codes have been generated. The generated XML file is placed res \ layout directory under the project. Then, it will be compiled with other files.

4.2. Android Embedded Java Files Generated

Right-click the Button widget, it pops up a programming interface event processing as shown in Figure 5. Then, writing the methods of event processing to the window which is for the methods of event processing. This part corresponds to the Event_Content which belongs to EventMode. After writing the code in accordance with the standard format, click the Save Button. So, event code of Button01 has been generated.

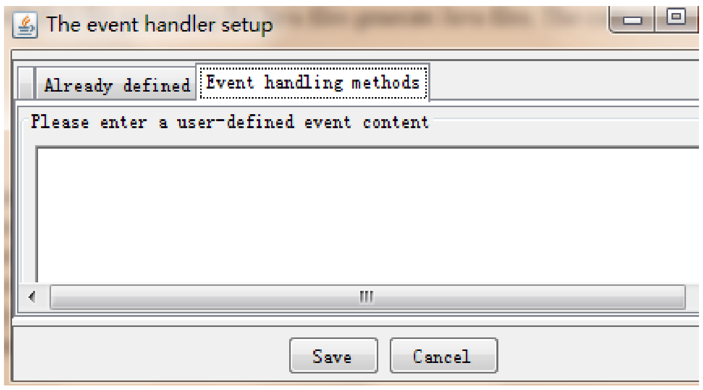


Figure 5. The Event Processing Settings Window

After the user finish setting the appropriate component of listen for events, according to the rules which are for generating the Java files generate Java files. The code is as follows:

```
package org.droiddraw;
import android.app.Activity;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
import android.widget.TextView;
public class DroidDrawActivity extends Activity{
private Button button31;
public void onCreate(Bundle savedInstanceState){
super.onCreate(savedInstanceState);
setContentView(R.layout.a1);
EditText01 = (EditText)findViewById( R.id.EditText01 );
TextView01 = (TextView)findViewById( R.id.TextView01 );
TextView02 = (TextView)findViewById( R.id.TextView02 );
Button01 = (Button)findViewById( R.id.Button01 );
Button01.setOnClickListener( new View.OnClickListener(){
        public void onClick(View v) {
        TextView02.setText( EditText01.getText().toString());
                                                  }
                });
 }}
```

From the experiment, we can get a conclusion that the method of code generation for Embedded Android based on the rule model has been basically achieved for the design of the embedded interface and automatically generate XML interface code and java application framework code. Users just need to drag and drop components and set its attributes. Then it can easily develop Android embedded applications. It can greatly improve the efficiency of development.

5. Conclusion

With the development of Android embedded operating system, the issue of Android embedded application development efficiency has been taken more attention by developers.

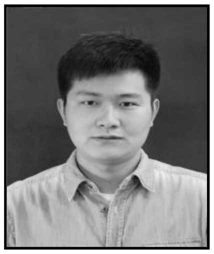
In this paper, the theory of rule model method achieves visual development for Android embedded application under the condition of less coding by defining a series of rules and models to constrain and control Android embedded application development and code automatically generation process. Compared with the traditional development mode, it greatly improves the development efficiency, reduces the development difficulty, and shortens the development cycle.

References

[1] T. Zhu and H. Li, "Review of applications based on Android.Computer& Telecommunication, vol. 01, no. 42, **(2011)**.
[2] H. Liu and Y. Xie, "Eclipse development platform and its application", Journal of Wuhan Automotive Polytechnic University, vol. 27, no. 02, 89, **(2005)**.
[3] X. Meng and Z. Huang, "The design of non-standard device driver for Android", Microcomputer & Its Applications, vol. 30, no. 9, **(2001)**.
[4] J. Heer and M. Agrawala, "Software Design Patterns for Information Visualization", IEEE Trans. on Visualization and Computer Graphics, Bangalore, India, **(2006)** December 6-8.
[5] W. Chen, W. Bian and Y. Zhang, "Porting LCD Drive Based on Android OS", Instrumentation Technology, vol. 14, no. 16, **(2014)**.
[6] Y. Wang and W. Chen, "Application Research on Advanced Model-driven Developing Method", Computer Engineering, vol. 32, no. 13, **(2006)**, pp. 63.
[7] H. Yang, R. Hou and Q. Tian, "Research on Model Supporting Interface Automatic Generation", Computer Engineering, vol. 36, no. 3, **(2012)**, pp. 79.
[8] J. Wang and Y. Zhu, "Model and constraints in Packing Problem", Journal Of Tianjin University And Education, vol. 21, no. 4, **(2011)**, pp. 1.
[9] Y. Zhang, Z. Li and Y. Wu, "Web content information extraction method based on rule model", Computer Engineering and Design, vol. 30, no. 20, **(2009)**, pp. 4665.
[10] H. Guo, "Android embedded application development Elaborates", Electronic Industry Publishers, Beijing, **(2010)**.

Authors

Wen Hu, master instructor, president of School of the Computer and Information Engineering, Harbin University of Commerce, backup leader of "Electronic Commerce" provincial key discipline echelon and academic leader of secondary doctoral discipline "electronic commerce and information service" in first-class doctoral discipline "business administration". His main research fields include computer network and communication, electronic commerce, embedded technology.



Kai Zhang, Master, student of School of the Computer and Information Engineering, Harbin University of Commerce. His main research fields include embedded technology.