2011 International Conference on Computer Science and Information Technology (ICCSIT 2011)
IPCSIT vol. 51 (2012) © (2012) IACSIT Press, Singapore
DOI: 10.7763/IPCSIT.2012.V51.129

Extracting Language Strings from XML Based on Android

Cheng Chen[+], Li Liu, Jianguo Chen, Cheng Zhang

School of Information Science and Engineering
Lanzhou University
Lanzhou, China

Abstract. Because of the rapid development of mobile phones, almost everyone can own a cell phone. Increasingly appearance of intelligent mobile phones makes competition more intense. The smartphone can support many languages now, so in this paper we present a method for extracting language strings from XML documents based on Android. The method implements an application of extracting language strings and saving them in an Excel file through parsing XML documents, and then rewrites XML documents from Excel. The running system shows that this method is effective and efficient.

Keywords: smartphone; Android; XML; parse

1. Introduction

Smartphone is such a class of mobile phones likes a personal computer as an independent operating system and users can install software, games and applications provided by the third-party service. Through such applications, it is possible to continue to expand the functionality of the phone and access networks via wireless mobile communications .In a number of intelligent machines, the use of android mobile operating system becomes more and more popular.

Android[1] was developed by Google and the Open Handset Alliance, which is a modern, open source operating system and a SDK(Software Development Kit) for mobile devices. With it users can create powerful mobile applications. Android is an open-source platform based on Linux, using java language in top-level application development. Java platform allows different ways to use XML (Extensible Markup Language) [2]. It is more convenient to use XML on the Internet.

To maintenance UI (User Interface) much easier and save more storage space, the resources (strings, graphics, icons, sound effects ...) displayed on Android phone screen are separated from the XML used to represent UI. Now because the mobile phones need to support multiple languages, the language translation work is also very important. Any single application is saved in the default resource file "res / values / strings.xml" in Android. The default strings are displayed in English on the phone screen. If languages of other countries are needed, the documents can be saved in the same level directory by a similar manner, for example, the strings contains Chinese are preserved in “res / values-zh-rCN/ strings.xml”. So when developers need strings of other languages in order to make it easier for the translator who can compare the default language and the language of strings, they need to extract the words string from XML documents, and then write them into an Excel file. For example, based on comparison of meanings of extracted Chinese and English strings, developers translate them into French. If developers use these strings to create applications, then developers should rewrite XML documents. These were subsequently stored in a translation of the data in Excel files, which can be extracted, then written in the original directory and file formats. Manual extraction of a single file wastes time greatly. How to complete the work efficiently is a problem that some mobile vendors need to address.

[+] Cheng Chen
ccccln@163.com

Handset manufacturers often modify strings shown on mobile screen in development. In this paper, we present a method to extracting language strings from XML documents based on Android. At first, we extract strings for translation or modifying by finding them file by file, and then we use XML parser to parse and rewrite XML by computer to save time. The target of parsing XML is to get nodes and the value of the nodes. People do researches on how to make parser more efficiency. But our goal is just to get strings under the first child node. We haven't considered extracting strings under each detail nodes. Therefore we implement an application and it is useful and efficient.

The paper is organized as follows. In section II, we introduce XML parsing technologies. The implement of the extracting method is given in section III. In section IV, we conclude the paper.

2. Parse xml

Language strings are preserved in XML documents, so we need to know how to parse XML documents. Java platform has supported many different ways to work with XML.

2.1. XML

XML is a general-purpose specification for creating custom markup languages. It is classified as an extensible language, because it allows the user to define the mark-up elements. XML's purpose is to aid information systems in sharing structured data, especially via the Internet, to encode documents, and to serialize data.

XML models data as a tree of elements. Arbitrary depth and width is allowed in such a tree, which facilitates storage of deeply nested data structures, as well as large collections of records or structures. Each element contains character data and can have attributes composed of name-value pairs. An XML document represents elements, attributes, character data, and the relationship between them by simply using angle brackets.

Figure 1 is an example of an XML document. The root node is named <Resources >. All other nodes in the document are contained within <Resources >.The root node holds two nodes, one is <string>, and the other is <plurals>. Node <string> contains text node "Zobrazit agendu" and element "name" with its value "show_agenda_view" , while node <plurals > holds two nodes <item>.

```
<resources xmlns:android=http://schemas.android.com/apk/res/android
    xmlns:xliff="urn:oasis:names:tc:xliff:document:1.2">
     <string name="show_agenda_view">"Zobrazit agendu"</string>
     <plurals name="Nhours">
        <item quantity="one">"1 hodina"</item>
        <item quantity="other">"<xliff:g id="COUNT">%d</xliff:g>hod."
        </item>
    </plurals>
</resources>
```

Figure 1. Example of XML.

2.2. Comparison of the Parser

XML DOM[3] is the short for W3C Document Object Model. It is a standard for accessing XML documents and it allows programs to dynamically access and update the XML document's content and structure. XML DOM actually defines the objects and properties of all XML elements, and the methods to access them. In other words, XML DOM is a standard for how to get, change, add, or delete XML elements. The XML DOM Parsing is thus a process mainly to load an XML document and construct such an XML DOM node tree in memory, so that parser functions can further access, insert, update, and delete DOM tree nodes.

SAX is a serial access parser API for XML. SAX provides a mechanism for reading data from an XML document. It is especially useful when handling streaming data and it is usually considered to be a much faster and less memory consuming way of processing XML since it does not need to parse the whole XML document into an in-memory DOM tree[4]. SAX parser is a parser which implements SAX functions as a

stream parser, with an event-driven API. The user can define a number of callback methods that will be called when events occur during parsing. The typical SAX parser has the events as the start and end of the XML document, the start and end of elements, and character data. XML attributes are provided as part of the data passed to the start element event. Events are fired when each of these are encountered. With this characteristic, SAX parsing is unidirectional, and previously parsed data cannot be re-read without starting the parsing process again.

In XML Pull Parser, rather than pushing events out to client code (SAX) or building a tree (DOM), it "pulls" events in by the application. Pull parsers must pull, parse and process all the elements in a document. Both XML Pull Parser and SAX parser represent the document as start tags, end tags, and comments as a sequence of events. They are similar, but in SAX, the parser (producer) drives the processing, while using the Pull Parser, the application (consumer) drives the processing. XML Pull Parser exposes a set APIs and an underlying set of events and through this way controls the application. In SAX, it calls user code by the parser when get the interesting part of XML input. The user code takes responsibility for keeping state between callbacks. In sum, the main difference is in pull parser, the user code controls the parsing process, and it can pull more data when it is ready to process.

DOM4J[5] is an open-source, object orientated alternative to DOM, and has many more features than its nearest rival, JDOM, which is, however, on the Java Community Process (JCP) standards track. The DOM4J APIs are very simple to use and provide a tree of java objects, which are relatively simple to convert in UWG gadget equivalents. It is also easy to manipulate this tree of objects when the hierarchy changes. It is also ideal for interactive applications because the entire object model is present in memory, where it can be accessed and manipulated by the user. To make more powerful, DOM4J use interfaces and abstract base class methods. DOM4J extensively uses the Collections API classes, but in many cases, it also provides a number of alternative methods to allow better performance or a more direct encoding method. Although DOM4J paid the cost of more complex API, it provides much more flexibility than JDOM. So we choose it for our application.

2.3. Parse Single XML Document

XML parsing process is to continue to read an XML document node by node, then the following elements and element property values. Example is demonstrated here of using SAX parser to illustrate the parsing method[6].

Still given the XML document in Figure 1, when parsing it with a SAX parser, the parser will generate the sequence of SAX events as: XML Element start, named resources =>attribute " xmlns:android " => attribute value "http://schemas.android.com/apk/res/android"=>attribute "xmlns:xliff"=>attribute value " urn:oasis:names:tc:xliff: document:1.2" => XML Element start, named string=> attribute "name"=>attribute value "show_agenda_view"=> XML Text node, with data equal "Zobrazit agendu"=>XML Element end named "string"=>……=>XML Element end named "resources".

Through this method, we know that computer how to read XML documents. Then we use DOM4J which is similar to the SAX to get the language strings. The methods written by DOM4J are easy to be used and overridden. Then we can implement our work.

3. Implement the application

This application mainly divided into two parts. The first part is to parse XML documents to get strings for writing into an Excel file. The second part is to write XML documents according to the strings in the Excel file. The flowchart of the application is shown in Figure 2.

3.1. Design

We need to get all strings displayed on the phone screen. For improving efficiency, we extract all strings from all applications XML documents which are under the directory of "app", so the point is the extraction of a single application solution[7]. Well shown in the Excel file, we extract required strings from the XML document then save them in the Excel file, each line of the string used to represent a default and the corresponding numbers of translated language strings, just as shown in Figure 3.

Hence, while extracting, we don't take detail to every child node, but the sub-node followed the root node. As an example, for an XML document shown in Figure 1, what we extract is strings like "Zobrazit agendu" and <item quantity="one">"1 hodina"</item> <item quantity="other"> "<xliff: g id="COUNT">%d</xliff: g> hod ."</item>.

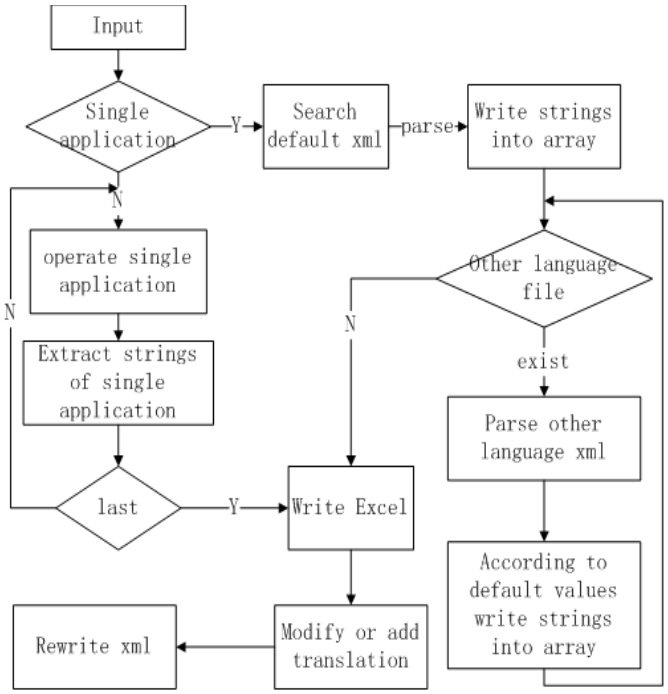

Figure 2. Flowchart of the application

Because English strings are the default strings, each application contains it at least. At first we extract English strings, and then according to the value of the second column shown in Figure 3, we insert other language strings behind.

After completing the work, we should consider using a language which is translated in the application. We write the translation into an Excel file. Then, the last work is how to rewrite XML on the basis of strings in Excel files. And the format of storage method is the same as parsing before.

3.2. Difficulties

To implement the application, we met some difficulties and the main problems are as follows:

Writing strings into an Excel file after parsing XML documents, it takes too long for us to insert other language strings comparing to the defaulted strings, and frequent errors happen.

There are many languages while parsing XML documents, we get language strings that show well under background, but if we write them into an Excel file, we find that the strings displayed incomplete, or code chaos.

Because we use the method named getText to extract strings, when we encounter strings like "<item quantity="one">"1 hodina"</item>", what we extract is null, then we find we can't extract right strings while there is "<".

After parsing XML documents, we extract right strings, then we rewrite XML from the Excel file, if we encounter strings just like "<item", then, what we see from rewritten XML is "<item". We find when we met angle brackets, the strings will change to special strings.

3.3. Solution

Since we will extract all strings, we save all values in a three-dimensional array, or we use a vector which contains a two-dimensional array. For single application, we save the strings in a two-dimensional array. Now we solve the difficulty, but new problem comes. After testing, we find that the using space of Excel is so large.

Checking it, we know the point is the size of the array. So we should analysis directory to get the numbers of language, then we can set the proper size.

For the second problem, first we thought the parsing process was wrong, then check the strings output streams and input streams, we focus on the writing code. We get characters byte by byte, and then find that different encoding formats lead to our error. So while writing Excel we set encoding format “ISO-8859-1”.

According to part II, we know XML is a special document. XML’s structure is like a tree. Under the tree there are nodes by nodes. We distinguish different nodes by special strings. If we only use the method named getText to extract values, we can’t extract the nodes, so null is shown. Analyzing parser current supported, we should get node by node while read XML documents. If we choose modifying source code, it will take too long. Considering strings shown in Excel line by line, so we extract the whole nodes as a string. The solution is first to find the beginning node which is just followed the first child node. Then we obtain the strings directly by giving up parsing.

Parsing XML document is the process of ongoing extract node. Similarly, writing XML is constantly process of ongoing create node. Computers cannot distinguish different nodes the same as people, they don't know how to distinguish which is a node or string value, so it need to use special characters to distinguish by parser in XML language. For example, when encounter "<" parser will regard it as one element which is the beginning to resolve. To avoid errors while parsing, the parser will change these special strings just like “<” to “<”, “>” to “>”, “&” to “&amp” and so on. So we prevent these strings escaping to new strings, if we do this, the problem is solved.

We have run the application on Eclipse, and we can extract all strings correctly from Android source efficiently. Also, we can rewrite XML from Excel the way we want.

4. Conclusion

XML has been developed for a long time, and there are also a lot of analytical methods for it. When developers encounter extraction of the language string, they will consider it to be a simple task. They may initially extract a single XML file, and finally find the rate of extraction very slow, and there are a lot of mistakes. So we put forward this method that can ensure the solution and the correctness of the XML file and can greatly improve work efficiency.

Smartphone is the mainstream of the development of mobile phone now, and the development of android is also very rapid. How to use network resources more effectively is one aspect of whether one can occupy the major market position after even using wired or wireless networks to achieve the call. After all, smartphone is not a computer, the use of mobile communications and network charging methods are different, so when connected to the network, or using a mobile communication, the data smartphone obtained should be special, the filtration technology[8] is needed. How to make more effective use of XML technology is an issue.

So in the future work, our research will focus on the filtering of network resources. It will make the resources which appear in smart phones more useful, not like popping up a lot of something useless and a waste of resources on the computer.

5. References

[1] Android Developers, “What is Android?”, Online at: http://developer.android.com/guide/basics/what-is-android.html. Last accessed:06/13/2010.

[2] W3C, “Extensible Markup Language (XML) 1.0 (Fifth Edition)”, http://www.w3. org/TR/2008/REC-xml-20081126/ (2008).

[3] Wei Xiaojuan, Ran Jing, Li AiHua,Yin ZhiBen, “XML Parse and Application Based on DOM,” Computer Technology and Development, vol.2, Apr.2007, pp.92-94, doi:CNKI:ISSN:1673-629X.0.2007-04-023.

[4] Zhao Junlan, “DOM and SAX Technology in XML Programming,” Computer Engineering, vol.24, 2004, pp.74-76, doi:CNKI:SUN:JSJC.0.2004-24-00R.

[5] Tan Kai,Yin Jinyu,Li Yafen, “Incremental Development Modeling Method of Web Application Based on DOM4J”, Modern Electronics Technique,vol.22,2009,pp.54-57,doi:CNKI:SUN:XDDJ.0.2009-22-015.

[6] Yinfei Pan, “Parallel XML Parsing”, Thesis, Graduate School of Binghamton University, State University of New York, 2009.

[7] M. Garofalakis, A. Gionis, R. Rastogi, S. Seshadri, K. Shim, “XTRACT: A System for Extracting Document Type Descriptors from XML Documents,”Proceeding of ACM SIGMOD international conference,vol 29,June.2000,pp.165-176,doi:10.1145/335191.335409.

[8] Hao Jin, “A Framework for Capturing, Querying, and Restructuring Metadata in XML Data,” Thesis, School of Electrical Engineering and Computer Science, August 2005.

Node na	name of value	English	German	Spanish	Italian	Japanese	Korean	Norwegian	Dutch	Polish	Russian	Chinese
string	app_label	Alarm Clock	"Wecker"	"Alarma"	"Sveglia"	"アラーム"	"알람 시계"	"Alarm"	"Wekker"	"Budzik"	"Будильник"	"闹钟时钟"
string	show_clock	Show clock	"Uhr anzeigen"	"Mostrar reloj"	"Mostra orologio"	"時計を表示"	"시계 표시"	"Vis klokke"	"Klok weergeven"	"Pokaż zegar"	"Показать часы"	"显示时钟"
string-ar	otherLabels	<item>Organization</item><item>N	<item>"Firma/Organisation"</ite	<item>"Organización"</item	<item>"Organizzazione"</item	<item>"所属"</item><item>"メ	<item>"조직"</item><item>"	<item>"Organisasjon"</item><item>"Notat	<item>"Organisatie"</item><ite	<item>"Organizacja"</item><item	<item>"Организация"</item>	<item>"组织"</item><item>"备
string	label	Label	"Label"	"Etiqueta"	"Etichetta"	"ラベル"	"라벨"	"Etikett"	"Label"	"Etykieta"	"Ярлык"	"标签"

Figure 3. Example of strings saved in Excel