LẬP TRINH TREN ANDROID

Part 3

Ví dụ 6. Trình xử lý SAX

```
import                                                  static
org.developerworks.android.BaseFeedParser.*;

public class RssHandler extends DefaultHandler{
    private List<Message> messages;
    private Message currentMessage;
    private StringBuilder builder;

    public List<Message> getMessages(){
        return this.messages;
    }
    @Override
    public  void  characters(char[]  ch,  int  start,  int
length)
            throws SAXException {
        super.characters(ch, start, length);
        builder.append(ch, start, length);
    }

    @Override
    public    void    endElement(String    uri,    String
localName, String name)
            throws SAXException {
        super.endElement(uri, localName, name);
        if (this.currentMessage != null){
            if (localName.equalsIgnoreCase(TITLE)){

currentMessage.setTitle(builder.toString());
            }                   else                    if
(localName.equalsIgnoreCase(LINK)){

currentMessage.setLink(builder.toString());
            }                   else                    if
```

```
(localName.equalsIgnoreCase(DESCRIPTION)){

currentMessage.setDescription(builder.toString());
            }                   else                    if
(localName.equalsIgnoreCase(PUB_DATE)){

currentMessage.setDate(builder.toString());
            }                   else                    if
(localName.equalsIgnoreCase(ITEM)){
                messages.add(currentMessage);
            }
            builder.setLength(0);
        }
    }

    @Override
    public void startDocument() throws SAXException {
        super.startDocument();
        messages = new ArrayList<Message>();
        builder = new StringBuilder();
    }

    @Override
    public   void   startElement(String   uri,   String
localName, String name,
            Attributes  attributes)  throws  SAXException
{
        super.startElement(uri,     localName,     name,
attributes);
        if (localName.equalsIgnoreCase(ITEM)){
            this.currentMessage = new Message();
        }
    }
}
```

Lớp `RssHandler` mở rộng lớp `org.xml.sax.helpers.DefaultHandler`. Lớp này cung cấp các thực thi mặc định, không thao tác cho tất cả các phương thức tương tự các sự kiện được tạo ra bởi trình phân tích SAX. Điều này cho phép các lớp con chỉ ghi chèn lên các

phương thức khi cần thiết. `RssHandler` có một API bổ sung, `getMessages`. Cái này trả về danh sách các đối tượng `Message` mà trình xử lý thu thập được khi nó nhận các sự kiện từ trình phân tích SAX. Nó có hai biến trong khác, một là `currentMessage` cho thể hiện `Message` đang được phân tích, và một là biến `StringBuilder` gọi là `builder` lưu trữ dữ liệu ký tự từ các nút văn bản. Các biến này đều được bắt đầu khi phương thức `startDocument` được dẫn ra khi trình phân tích gửi sự kiện tương ứng cho trình xử lý.

Hãy xem phương thức `startElement` trong Ví dụ 6. Phương thức này được gọi mỗi khi bắt gặp thẻ mở trong tài liệu XML. Bạn chỉ cần quan tâm khi nào thẻ đó là thẻ `ITEM`. Trong trường hợp đó, bạn tạo ra một `Message` mới. Bây giờ hãy nhìn vào phương thức `characters`. Phương thức này được gọi ra khi bắt gặp dữ liệu ký tự từ các nút văn bản. Dữ liệu dễ dàng được thêm vào biến `builder`. Cuối cùng hãy xem phương thức `endElement`. Phương thức này được gọi ra khi bắt gặp thẻ kết thúc. Đối với các thẻ tương ứng với các đặc tính của một `Message`, giống như `TITLE` và `LINK`, đặc tính thích hợp được thiết đặt trên `currentMessage` sử dụng dữ liệu từ biến `builder`. Nếu thẻ kết thúc là một `ITEM`, thì `currentMessage` thêm vào danh sách Messages. Đây là sự phân tích SAX rất điển hình; ở đây không có gì là duy nhất đối với Android. Vì thế nếu bạn biết cách viết một trình phân tích SAX Java, thì bạn biết cách viết một trình phân tích SAX Android. Tuy nhiên, Android SDK có bổ sung thêm một số tính năng thuận tiện vào SAX.

Phân tích SAX dễ dàng hơn

Android SDK có chứa một lớp tiện ích được gọi là `android.util.Xml`. Ví dụ 7 trình bày cách cài đặt một trình phân tích SAX với cùng lớp tiện ích như thế.

Ví dụ 7. Trình phân tích SAX Android

```
public class AndroidSaxFeedParser extends
BaseFeedParser {

    public AndroidSaxFeedParser(String feedUrl) {
        super(feedUrl);
    }
```

```
    public List<Message> parse() {
        RssHandler handler = new RssHandler();
        try {
            Xml.parse(this.getInputStream(),
Xml.Encoding.UTF_8, handler);
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
        return handler.getMessages();
    }

}
```

Lưu ý là lớp này vẫn sử dụng trình xử lý SAX chuẩn, vì đơn giản bạn đã sử dụng lại `RssHandler` như trong Ví dụ 7 ở trên. Việc có thể sử dụng lại trình xử lý SAX rất tốt, nhưng nó vẫn có đôi chút phức tạp về mã trình. Bạn có tưởng tượng, nếu bạn phải phân tích một tài liệu XML phức tạp hơn rất nhiều, trình phân tích có thể trở thành mảnh đất màu mỡ cho các lỗi. Ví dụ, hãy xem lại phương thức `endElement` trong Ví dụ 6. Lưu ý cách phương thức này kiểm tra như thế nào nếu `currentMessage` có giá trị không trước khi nó cố cài đặt các thuộc tính? Bây giờ hãy nhìn vào XML mẫu trong Ví dụ 4. Lưu ý rằng có các thẻ `TITLE` và `LINK` nằm ngoài các thẻ `ITEM`. Đó là lý do tại sao kiểm tra giá trị không được đưa vào. Nếu không thì thẻ `TITLE` đầu tiên có thể gây ra một `NullPointerException`. Android bao gồm cả biến thể SAX API của chính nó (xem Ví dụ 8) loại bỏ yêu cầu bạn phải viết trình xử lý SAX của chính bạn.

Ví dụ 8. Trình phân tích SAX Android đơn giản

```
public class AndroidSaxFeedParser extends
BaseFeedParser {

    public AndroidSaxFeedParser(String feedUrl) {
        super(feedUrl);
    }

    public List<Message> parse() {
        final Message currentMessage = new Message();
```

```
        RootElement root = new RootElement("rss");
        final    List<Message>    messages    =    new
ArrayList<Message>();
        Element channel = root.getChild("channel");
        Element item = channel.getChild(ITEM);
        item.setEndElementListener(new
EndElementListener(){
            public void end() {
                messages.add(currentMessage.copy());
            }
        });

item.getChild(TITLE).setEndTextElementListener(new
EndTextElementListener(){
            public void end(String body) {
                currentMessage.setTitle(body);
            }
        });

item.getChild(LINK).setEndTextElementListener(new
EndTextElementListener(){
            public void end(String body) {
                currentMessage.setLink(body);
            }
        });

item.getChild(DESCRIPTION).setEndTextElementListener(ne
w
EndTextElementListener(){
            public void end(String body) {
                currentMessage.setDescription(body);
            }
        });

item.getChild(PUB_DATE).setEndTextElementListener(new
EndTextElementListener(){
            public void end(String body) {
                currentMessage.setDate(body);
            }
        });
```

```
        try {
            Xml.parse(this.getInputStream(),
Xml.Encoding.UTF_8,
root.getContentHandler());
        } catch (Exception e) {
            throw new RuntimeException(e);
        }
        return messages;
    }
}
```

Như đã hứa, mã phân tích SAX mới không sử dụng trình xử lý SAX. Thay vào đó nó sử dụng các lớp từ gói android.sax trong SDK. Các lớp này cho phép bạn mô hình hóa cấu trúc của tài liệu XML của bạn và thêm một trình nghe sự kiện nếu cần. Trong mã trình trên, bạn khai báo rằng tài liệu của bạn sẽ có một phần tử gốc có tên `rss` và rằng phần tử này sẽ có ba phần tử con là `channel`. Tiếp đến bạn nói rằng `channel` sẽ có ba phần tử con được gọi là `ITEM` và bạn bắt đầu gắn các trình nghe. Đối với mỗi trình nghe, bạn đã sử dụng một lớp bên trong vô danh đã thực hiện giao diện bạn quan tâm (hoặc `EndElementListner` hoặc `EndTextElementListener`). Chú ý không cần phải theo dõi dữ liệu ký tự. Việc này không chỉ đơn giản hơn mà thực sự còn hiệu quả hơn. Cuối cùng, khi bạn gọi dẫn phương thức tiện ích Xml.parse, bây giờ bạn đưa vào trình xử lý được tạo ra từ phần tử gốc.

Toàn bộ mã trình ở trên trong Ví dụ 8 thuộc loại tùy chọn. Nếu bạn thấy thoải mái với mã trình phân tích SAX chuẩn trong môi trường Java, thì bạn có thể tích vào đó. Nếu bạn muốn thử các trình bao bọc tiện lợi do Android SDK cung cấp, bạn cũng có thể sử dụng nó. Nếu bạn không muốn sử dụng SAX thì sao đây? Vẫn còn có một vài lựa chon khác. Lựa chọn đầu tiên bạn sẽ thấy đó là DOM.
