分散システム

第六回レポート課題

課題 1

【課題内容】

講義資料（その 3）ページ 18,19 に記載されているソースコードを参考に、SAX API を用いて XML 文書のすべての要素や属性に関する情報を出力するプログラムを作成せよ。ただし、

- 作成したソースコード
- 読み込ませた XML 文書ファイル（小さいものでもよい）
- 実行結果（一部でもよい。スクリーンダンプにより）

を示すこと。また、実行結果には、実行した時刻を出力すること。

【作成したソースコード】

≪TraceEvents.java≫

```
import javax.xml.parsers.*;

public class TraceEvents {
    public static void main(String[] args) {
        System.out.println("current time = " + new java.util.Date());
        try {
            /* XML プロセッサの生成 */
            SAXParserFactory factory = SAXParserFactory.newInstance();
            SAXParser parser = factory.newSAXParser();
            TraceEventsHandler handler = new TraceEventsHandler();

            /* 構文解析の実行 */
            parser.parse(args[0],handler);
        } catch(Exception e) {
            System.out.println("Exception occurred: " + e.getMessage());
        }
    }
}
```

≪TraceEventsHandler.java≫

```
import org.xml.sax.*;
import org.xml.sax.helpers.DefaultHandler;

public class TraceEventsHandler extends DefaultHandler {
    public TraceEventsHandler() {}

    public void startDocument() throws SAXException {
        // Document 開始時の処理
        System.out.println ( "[startDocument] : " );
    }

    public void endDocument() throws SAXException {
        // Document 終了時の処理
        System.out.println ( "[endDocument]    : " );
    }

    public void startElement(String uri, String localpart, String name, Attributes amap) {
        // Element 開始時の処理
        System.out.println ( "<startElement>   : uri = " + uri + ", localpart = " + localpart + ",
name = " + name );
        for(int i=0;i<amap.getLength();i++) { /* 属性に対する処理 */
            System.out.println("   attribute name = " + amap.getQName(i) + ", type = " +
amap.getType(i) + ", value = " + amap.getValue(i));
        }
    }

    public void endElement(String uri, String localName, String qName) throws
SAXException {
        // Element 終了時の処理
        System.out.println ( "<endElement>     : uri = " + uri + ", localName = " +
localName + ", qName = " + qName );
    }

    public void characters(char[] ch, int start, int length) {
        // Text 情報の受け取り
        String text = new String(ch, start, length);
        System.out.println ( "(characters)     : " + text);
    }
}
```

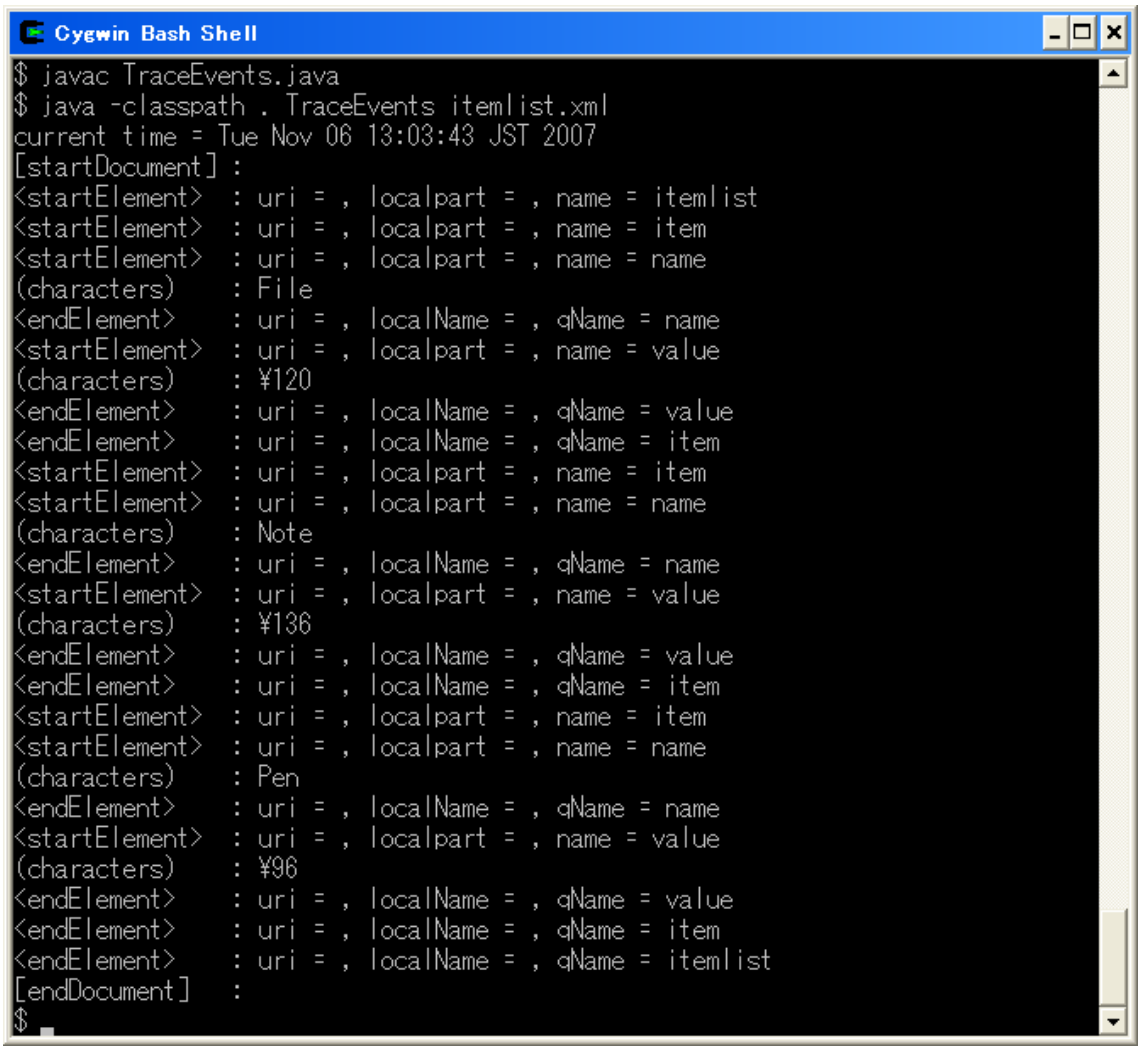
【読み込ませた XML 文書ファイル】

≪itemlist.xml≫

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE ItemList SYSTEM "itemlist.dtd">

<itemlist>
    <item>
        <name>File</name>
        <value>¥120</value>
    </item>
    <item>
        <name>Note</name>
        <value>¥136</value>
    </item>
    <item>
        <name>Pen</name>
        <value>¥96</value>
    </item>
</itemlist>
```

【実行結果】


Cygwin Bash Shell
$ javac TraceEvents.java
$ java -classpath . TraceEvents itemlist.xml
current time = Tue Nov 06 13:03:43 JST 2007
[startDocument] :
<startElement> : uri = , localpart = , name = itemlist
<startElement> : uri = , localpart = , name = item
<startElement> : uri = , localpart = , name = name
(characters)   : File
<endElement>   : uri = , localName = , qName = name
<startElement> : uri = , localpart = , name = value
(characters)   : ¥120
<endElement>   : uri = , localName = , qName = value
<endElement>   : uri = , localName = , qName = item
<startElement> : uri = , localpart = , name = item
<startElement> : uri = , localpart = , name = name
(characters)   : Note
<endElement>   : uri = , localName = , qName = name
<startElement> : uri = , localpart = , name = value
(characters)   : ¥136
<endElement>   : uri = , localName = , qName = value
<endElement>   : uri = , localName = , qName = item
<startElement> : uri = , localpart = , name = item
<startElement> : uri = , localpart = , name = name
(characters)   : Pen
<endElement>   : uri = , localName = , qName = name
<startElement> : uri = , localpart = , name = value
(characters)   : ¥96
<endElement>   : uri = , localName = , qName = value
<endElement>   : uri = , localName = , qName = item
<endElement>   : uri = , localName = , qName = itemlist
[endDocument]   :
$

課題 2

【課題内容】

第五回の課題 2 と同様の機能（すなわち、商品データの XML 文書を用いて、指定された商品名に対応する価格を検索し、出力する）を、SAX API を用いて構築せよ。ただし、

- 作成したソースコード
- 読み込ませた XML 文書ファイル（小さいものでもよい）
- 実行結果（一部でもよい。スクリーンダンプにより）

を示すこと。また、実行結果には、実行した時刻を出力すること。

【作成したソースコード】

≪SAXSearch.java≫

```
import javax.xml.parsers.*;

public class SAXSearch {
    public static void main(String[] args) {
        System.out.println("current time = " + new
java.util.Date(System.currentTimeMillis()));
        try {
            /* XML プロセッサの生成 */
            SAXParserFactory factory = SAXParserFactory.newInstance();
            factory.setValidating(true);
            SAXParser parser = factory.newSAXParser();
            SAXSearchHandler handler = new SAXSearchHandler(args[1]);
            /* 構文解析の実行 */
            parser.parse(args[0],handler);
        } catch(Exception e) {
            System.out.println("Exception occurred: " + e.getMessage());
        }
    }
}
```

≪SAXSearchHandler.java≫

```
import org.xml.sax.*;
import org.xml.sax.helpers.DefaultHandler;

public class SAXSearchHandler extends DefaultHandler {
    String target; /* 検索対象の商品名 */
    String itemName; /* 商品名を保管する変数 */
    String value;          /* 価格を保管する変数 */

    boolean insideNameElem = false; /* 商品名の要素に入っていることを示すフラグ変数 */
    boolean insideValueElem = false; /* 価格の要素に入っていることを示すフラグ変数 */
    boolean isfind = false;          /* 見つかったかどうかを示すフラグ変数 */

    public SAXSearchHandler(String target) {
        super();
        this.target = target;
    }

    public void startDocument() throws SAXException {
        // Document 開始時の処理
    }

    public void endDocument() throws SAXException {
        // Document 終了時の処理
    }

    public void startElement(String uri, String localpart, String name, Attributes amap) {
        // Element 開始時の処理
        if(name.equals("itemlist")) {
            /* ルート要素に対する処理 */
        } else if(name.equals("name")) {
            /* 商品名要素に対する処理 */
            insideNameElem = true;
        } else if(name.equals("value")) {
            /* 価格要素に対する処理 */
            insideValueElem = true;
        }
    }

    public void endElement(String uri, String localName, String qName) throws
SAXException {
        // Element 終了時の処理
        if(qName.equals("item")) {
            /* 商品要素が閉じられたときの処理 */
            if ( itemName.equals(target) ) {
                System.out.println( itemName + "の価格は" + value   + "です。" );
                isfind = true;
            }
        } else if(qName.equals("name")) {
            /* 商品名要素が閉じられたときの処理 */
            insideNameElem = false;
        } else if(qName.equals("value")) {
            /* 価格要素が閉じられたときの処理 */
            insideValueElem = false;
        } else if(qName.equals("itemlist")) {
            /* ルート要素が閉じられたときの処理 */
            if ( !isfind ) {
                System.out.println( target + "は見つかりませんでした。" );
            }
        }
    }
```

```
    public void characters(char[] ch, int start, int length) {
        // Text 情報の受け取り
        String text = new String(ch,start,length);
        /* 商品名や価格の保存 */
        if ( insideNameElem ) {
            itemName = text;
        }
        if ( insideValueElem ) {
            value = text;
        }
    }
}
```

【読み込ませた XML 文書ファイル】

≪itemlist.xml≫

```
<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE ItemList SYSTEM "itemlist.dtd">

<itemlist>
    <item>
        <name>File</name>
        <value>¥120</value>
    </item>
    <item>
        <name>Note</name>
        <value>¥136</value>
    </item>
    <item>
        <name>Pen</name>
        <value>¥96</value>
    </item>
</itemlist>
```

【実行結果】